SCI INSTITUTE
TECHNICAL REPORT

Interactive Exploration of Volumetric Data Sets With a Combined Visual and Haptic Interface

Milan Ikits

UUSCI-2007-005

Scientific Computing and Imaging Institute
University of Utah
Salt Lake City, UT 84112 USA

April 25, 2007

Abstract:

The analysis of complex volumetric data sets is a critical component of many scientific and engineering applications. The difficulty of understanding the increasing amount of data generated by these applications motivates the need for effective and intuitive visualization approaches that allow users to extract relevant information from the data in a relatively short amount of time. Even though a great variety of visualization techniques are available today, the effectiveness of a particular approach is often limited by the ability of the user to interact with the presented information. Incorporating three-dimensional interaction metaphors into the visualization environment has great potential to assist the user during the data exploration process.

This dissertation develops and evaluates interactive visualization methods that combine three-dimensional graphical and haptic displays for the exploration of volumetric data sets. A number of techniques are suggested that are suitable for augmenting the traditional visualization interface with visual and haptic interaction methods for guiding the user during exploration. The advantage of the presented approach is that the rendering algorithms do not require a preprocessing stage to extract intermediate geometric representations from the data. Thus, the user has direct control over the visual and haptic display via a set of visualization parameters.

In addition to demonstrating the effectiveness of the presented methods over previous approaches, the dissertation describes a controlled experiment that examines the contribution of haptic guidance to vector field exploration tasks. The results of the evaluation suggest that properly combined visual and haptic feedback can significantly improve the speed and accuracy of volumetric data exploration. The presented methods are further illustrated with examples of representative visualization applications.

THE UNIVERSITY OF UTAH

INTERACTIVE EXPLORATION OF VOLUMETRIC DATA SETS WITH A COMBINED VISUAL AND HAPTIC INTERFACE

by

Milan Ikits

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

School of Computing

The University of Utah

May 2007

THE UNIVERSITY OF UTAH GRADUATE SCHOOL

SUPERVISORY COMMITTEE APPROVAL

of a dissertation submitted by

Milan Ikits

This dissertation has been read by each member of the following supervisory committee and by majority vote has been found to be satisfactory.

	Chair: Charles D. Hansen
	Christopher R. Johnson
	John M. Hollerbach
	Peter S. Shirley
	Robert S. MacLeod

THE UNIVERSITY OF UTAH GRADUATE SCHOOL

FINAL READING APPROVAL

To the Graduate Council of the University of Utah:

I have read the dissertation of Milan Ikits in its final form and have found that (1) its format, citations, and bibliographic style are consistent and acceptable; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the Supervisory Committee and is ready for submission to The Graduate School.

Date

Charles D. Hansen
Chair: Supervisory Committee

Approved for the Major Department

Martin Berzins
Chair/Director

Approved for the Graduate Council

David S. Chapman
Dean of The Graduate School

ABSTRACT

The analysis of complex volumetric data sets is a critical component of many scientific and engineering applications. The difficulty of understanding the increasing amount of data generated by these applications motivates the need for effective and intuitive visualization approaches that allow users to extract relevant information from the data in a relatively short amount of time. Even though a great variety of visualization techniques are available today, the effectiveness of a particular approach is often limited by the ability of the user to interact with the presented information. Incorporating three-dimensional interaction metaphors into the visualization environment has great potential to assist the user during the data exploration process.

This dissertation develops and evaluates interactive visualization methods that combine three-dimensional graphical and haptic displays for the exploration of volumetric data sets. A number of techniques are suggested that are suitable for augmenting the traditional visualization interface with visual and haptic interaction methods for guiding the user during exploration. The advantage of the presented approach is that the rendering algorithms do not require a preprocessing stage to extract intermediate geometric representations from the data. Thus, the user has direct control over the visual and haptic display via a set of visualization parameters.

In addition to demonstrating the effectiveness of the presented methods over previous approaches, the dissertation describes a controlled experiment that examines the contribution of haptic guidance to vector field exploration tasks. The results of the evaluation suggest that properly combined visual and haptic feedback can significantly improve the speed and accuracy of volumetric data exploration. The presented methods are further illustrated with examples of representative visualization applications.

To my family

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

A few words are not sufficient to acknowledge all the people who supported me during my graduate studies. I would like to thank my advisor, Chuck Hansen, for providing funding, equipment, and guidance for accomplishing the research culminating in this dissertation. John Hollerbach taught me about the importance of scientific rigor and provided support during the early years of my graduate studies. I also thank my other committee members, Chris Johnson, Pete Shirley, and Rob MacLeod, for the constructive feedback on my work.

I am grateful to many of my colleagues at the University of Utah who collaborated with me over the past several years. In particular, J. Dean Brederson deserves the credit for leading the design and construction of the Visual Haptic Workbench. Dean and I had numerous conversations about research topics on calibration, haptic rendering, and human-computer interaction, which led to the development of several techniques presented in the dissertation. I am also thankful to Sarah Creem-Regehr from the Department of Psychology at the University of Utah for helpful discussions on the design of the user study and providing access to statistical analysis tools for processing the experimental data. Gordon Kindlmann, myself, and several hundred cups of coffee spent many hours on enlightening and refreshing discussions about a variety of research topics over the years. In addition, Gordon created one of the figures in the dissertation and provided useful feedback on this work. My continuing interest in hardware-assisted volume rendering and data processing is attributed to collaborations with Joe Kniss, Aaron Lefohn, Simon Premože, Emil Praun, Steve Callahan, João Comba, and Cláudio Silva. I feel privileged to have worked with Dave Weinstein and Greg Jones, who provided me with an opportunity to contribute to the success of a commercial medical visualization application.

I have made many friends along the way, who helped me, directly or indirectly, in my struggle to complete the requirements for the Ph.D. program. I would like to thank Mike Stark, Tushar Udeshi, Don Nelson, Betty Mohler, Miriah Meyer, Hope Eksten, Rose Mills, Vegard Lund, Tim Miller, Frank Stenger, Chris Wyman, Shaun Ramsey, Richard Coffey,

Nathan Galli, Erik Jorgensen, Chad Lake, Mike Parker, Sean Heffernan, and Ramy Sadek for their help, feedback, humor, and encouragement. I am especially grateful to Miriah and Betty for their help with the dissertation document and the defense presentation.

The dissertation could not have been completed without the support of my family. I especially owe my sanity to my wife, Bettina, and my son, Mendi, for their patience, love, and encouragement during all the challenges and difficulties I have faced. I am grateful to my parents, Tamás and Katalin, who believed in me and supported my education from the very beginning. I would also like to thank the Alsops, the Roses, and the Morgans for their invaluable help with settling down in Utah so far away from home.

Financial support for this research was provided by the National Science Foundation under grant ACI-9978063 and the Department of Energy Advanced Visualization Technology Center. The Teem toolkit [89] was used for data probing and processing in this work. The computational fluid dynamics data set is provided courtesy of J. Bell and V. Beckner, Center for Computational Sciences and Engineering, Lawrence Berkeley National Laboratory. The diffusion tensor data set is courtesy of G. Kindlmann and A. Alexander, W. M. Keck Laboratory for Functional Brain Imaging and Behavior, University of Wisconsin-Madison. The heart data set is courtesy of P. Hunter, Bioengineering Institute, University of Auckland.

CHAPTER 1

INTRODUCTION

The analysis of complex volumetric data sets is a critical component of many scientific and engineering disciplines, such as computational fluid dynamics, medical image analysis, and computer-aided design and manufacturing. The difficulty of handling the increasing amount of data generated and used by scientific and engineering applications motivates the need for more efficient and intuitive visualization approaches that allow users to extract relevant information from the data in a relatively short amount of time.

Even though a great variety of graphical visualization techniques have been developed in the past, effective presentation of complex multidimensional and multifield data sets remains a challenging problem. The effectiveness of a particular visualization approach is often limited by the ability of the expert user to interact with the presented information. When examining objects in real and synthetic worlds, humans predominantly rely on the visual channel. Although the human visual system is excellent at interpreting two-dimensional patterns and images, understanding complex volumetric features is difficult due to occlusion, clutter, and lack of spatial cues. To obtain further information or to disambiguate existing visual cues from the environment, humans frequently utilize other sensory input. Examples include the use of proprioception while simultanously obtaining multiple views to determine the relative location and orientation of surronding objects, or the use of the kinesthetic and tactile senses to gather additional information about the shape and material properties of a particular object of interest. Since humans interact with spatial information most naturally in three dimensions, incorporating real-world interaction metaphors into a computer generated environment has great potential to assist with the data exploration process. This dissertation develops and evaluates visualization methods that combine three-dimensional graphical and haptic displays for the exploration of multidimensional and multifield volumetric data sets.

1.1 Global and Local Visualization

Users of visualization applications are interested in extracting information from the data at global and local levels of detail. The global perspective is required for navigating the data space to locate possible features of interest in the data. Usually, the global level provides an overall view of the data and is often presented in a simplified or reduced form. For most data sets it is not possible to create a global view that captures all the necessary information without cluttering the data space. One example is vector field visualization, where the goal is to convey large amounts of three-dimensional directional and magnitude information. When using a global operation, such as drawing a vector glyph at every data cell, it becomes very difficult to discern features due to the occlusion and clutter caused by the large number of glyphs displayed. Sometimes it can be very difficult to observe any correspondence between features from a single global image of the data. Thus, the global view is often augmented with a local perspective to assist with meaningful, precise, and detailed information extraction. The local visualization is frequently under interactive control by the user. One example is the rake probe used in vector field visualization applications. Using the probe the operator can seed a small number of streamlines in the field. As shown in Figure 1.1, when the rake probe is used, the local field structure is captured adequetly, but the global context is lost. By moving the probe around, users can quickly explore the relevant features, such as vortices, in the data. Even though for vector fields it is possible to extract the features and present them in a simplified global view, many users prefer the more direct and intuitive data probing approach. Other examples of local visualization methods include clipping and slicing tools, which make the displayed information easier to comprehend by restricting it to a selected region of the data space.

Many global and local visualization techniques are developed in isolation for a specific scalar, vector, or tensor data set. Often, however, researchers would like to explore the interaction between multiple field variables using intermixed visualization techniques. The combined use of global and local visualization methods has proven valuable for the exploration of multifield and multidimensional data sets, for which it is impossible to present all available information in a single global view. For example, as shown in Figure 1.1, a volume rendered global view of a scalar field can be effectively combined with a streamline representation of a vector field, resulting in a global composite visualization

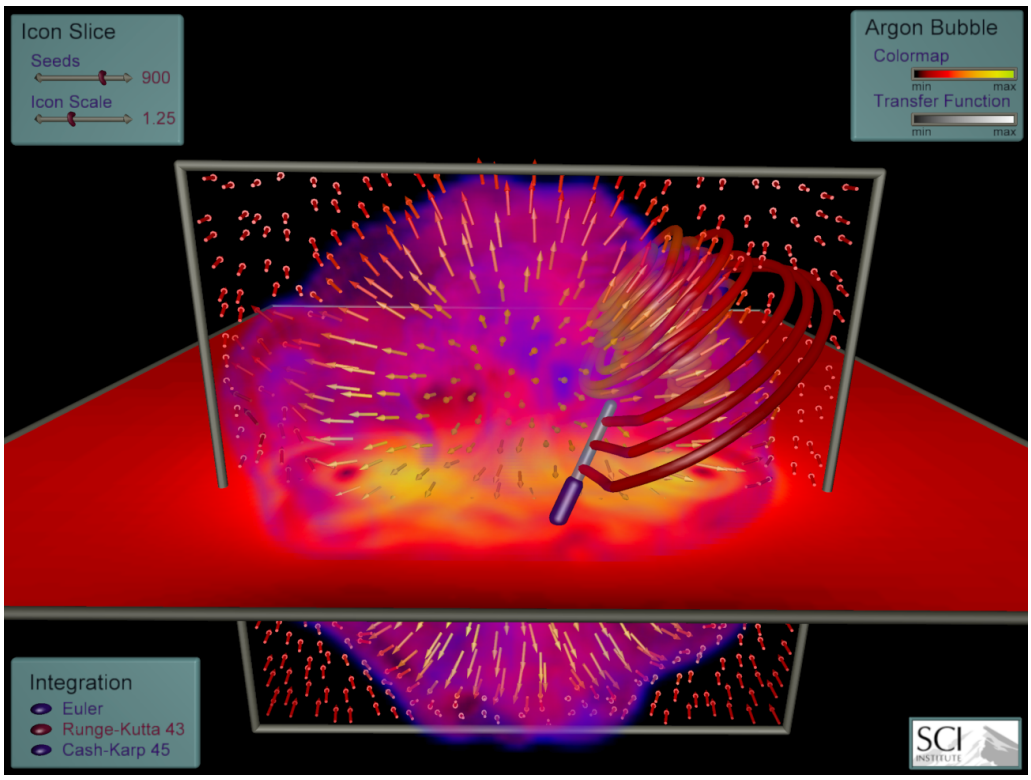


Figure 1.1. Interactive exploration of a computational fluid dynamics data set. Global context is provided by volume rendering the scalar component of the data. The global view is augmented with local data exploration tools to help users find important features in the data.

of the quantities of interest. The local probe can provide detailed specific information about the interaction of the two variables without cluttering the view.

The combined use of global and local visualization techniques does not provide a complete solution to the fundamental problem of the limited information processing capacity of the human visual system. While local visualization techniques are effective at revealing selected features in the data, it is easy to miss interesting phenomena when using exploratory visualization tools. In addition, the limited spatial resolution of the computer screen restricts the number of graphical cues that can be discerned at a time [169]. Stereoscopic rendering, shadows, and proper illumination provide depth cues that make volumetric feature recognition and discrimination more accurate. Incorporating transparency into the rendering model reduces occlusion and clutter at the price of increasing the ambiguity of the visualization. Finally, interactive tools that provide guidance to

the user during exploration are either not available or cumbersome to use due to the 2D nature of the desktop user interface.

1.2 Immersive Environments

Immersive virtual reality systems place the user in a computer generated environment that mimicks characteristics and behavior of the real world. The primary advantages of immersive environments over desktop systems include a rich set of spatial cues provided by head-tracked stereoscopic viewing and the possibility for human-centric interaction via tracked input devices and additional sensory feedback. Immersive manipulation and exploration techniques frequently borrow analogues from real-world experience. One example is picking up an object of interest in one hand and examining it with a magnifying glass held in the other hand. Such assymetric two-handed interaction techniques are often incorporated into immersive visualization applications [63, 61, 42, 37]. In general, immersive environments allow users to more actively and efficiently participate in the data exploration process through natural interaction metaphors.

Immersive environments have been proposed for a variety of volume visualization and modeling tasks. The tasks typically involve positioning and manipulating interface objects within the workspace using 3D input devices. Examples range from direct manipulation of visualization tools such as cutting planes or data probes [123, 61] to building geometric models from the data [160, 61]. Figure 1.2 shows an expert using tracked spatial input devices to manipulate a rake widget inside a large-scale stereoscopic visualization of a data set. The advantage of spatial input devices is that the user does not need to break down the task to a sequence of 2D operations. Consequently, tasks that require constant hand-eye coordination are better suited to immersive environments [64].

Immersive environments and applications present a number of engineering challenges. One of the grand challenges of using immersive environments for scientific visualization is making user interaction comfortable, fast, and effective [171]. Interaction with a computer-generated 3D world is difficult, because many cues, constraints, and affordances that are part of the real world are missing from the virtual world [14]. In contrast, desktop interaction has been very successful due to its simplicity, robustness, and convenience. In the past, a variety of methods have been proposed to make immersive interaction more effective. For example, props are task-specific passive input devices that provide additional kinesthetic cues for more intuitive object manipulation [63, 154]. Physically-based

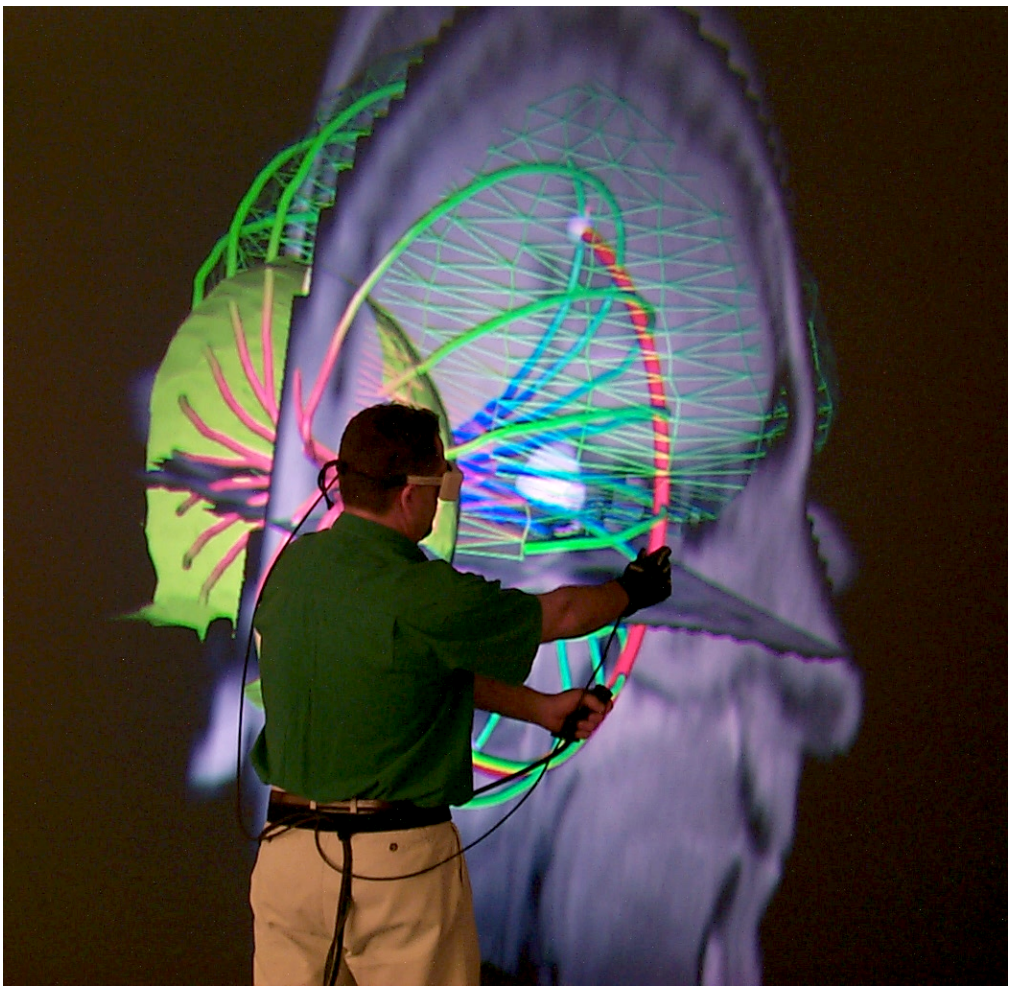


Figure 1.2. Example of an immersive data exploration environment. An expert user interacts with a large-scale stereoscopic visualization of a volumetric data set. The user can specify visualization parameters via tracked spatial input devices.

manipulation tools can help make interaction with virtual objects more natural [51, 96]. Finally, haptic feedback can provide programmable environment constraints to facilitate precise manipulation of the user interface components.

1.3 Haptic Interfaces

Haptic feedback is a promising interaction modality for a variety of applications. Successful commercial examples of haptic feedback include virtual prototyping tools [159], game controllers [111], and medical training simulators [81]. Graphical applications are augmented with force or tactile feedback for two reasons: 1. to increase the realism of

the simulation and 2. to improve operator performance in terms of increased precision, reduced fatigue, and shorter task completion times.

The primary advantage of haptic rendering is that it provides a bidirectional flow of information via position sensing and force feedback [2, 20]. The coupled information flow creates possibilities for more natural and intuitive interaction and exploits additional sensory channel bandwidth of the user. When users are presented with a proper combination of visual and haptic information, they experience a sensory synergy resulting from the physiological reinforcement of the displayed multimodal cues [41, 136].

Incorporating haptic feedback into visualization applications has been reported to aid in the understanding of complex scientific data sets and enables the development of fluid human computer interfaces [95, 172]. As an additional source of input to the user, the haptic channel can assist the visual channel in the task of comprehending many aspects of the data simultaneously. In addition, haptic cues can reinforce or disambiguate the shape and depth of objects in the scene. Experiments have shown that user performance increases for a simple tapping task when haptic feedback is added to either a monoscopic [176] or a stereoscopic display [4]. Even though force feedback has a positive overall contribution to depth perception [13], quantifying the contribution is not always straightforward, because the feedback frequently alters the characteristics of the task [4, 177].

In virtual environment applications, haptic feedback has been primarily used for augmenting 2D and 3D user interfaces [124, 134, 131, 95]. In this context the role of force feedback is to enhance the interface by composing haptic effects that fit into one or more of the following categories [124]:

- Anticipation: Haptic effects that allow users to anticipate the outcome of their input, such as an increasingly resisting force when pushing a virtual button.
- Reinforcement: Effects indicating that an action has been completed, such as snapping the cursor to a grid point.
- Guidance: Feedback that restricts user input to a set of possible outcomes. For example, a line constraint improves the precision and speed of sketching straight curve segments in space.
- Indication: Effects indicating that a certain condition is active, such as using haptic textures for displaying icons on a desktop interface.

The above categorization of haptic effects is also applicable to data probing tasks. For example, one way to provide feedback for the exploration of vector field data is to constrain the motion of the probe along a selected streamline such that any effort to move the probe in a direction perpendicular to the current orientation of the field results in an increasingly opposing force. Such feedback is a form of guidance, because it restricts user input to a single field line only. It is also an efficient indication of the local vector field orientation. As shown in Chapter 7, guidance can substantially reduce the mental effort and workload needed for spatial data exploration tasks. When the user pushes the probe hard enough perpendicular to the local field orientation, the probe can "pop over" to an adjacent streamline, providing both anticipatory and reinforcing feedback. Such interaction allows the user to move the probe in arbitrary directions and still receive strong haptic cues about the characteristic features of the flow field. Finally, an additional force component can be used along the streamline to indicate the magnitude of the field. Alternatively, a secondary constraint can be added to convey information about the speed of the flow in the form of haptic tickmarks.

The above application of haptic user interface concepts differs significantly from the approach of mapping data variables to force and torque components [84, 40]. Since very few haptic devices provide more than 3DOF feedback to the user, the data mapping approach quickly runs into limitations. It is also not clear how to combine multiple data modalities in a meaningful way [84]. Even if additional degrees of freedom are available, they are not necessarily useful for data exploration tasks. Torque feedback is essential in simulation applications, where the goal is to provide a realistic environment to the user. For example, there would be no use for a mechanical assembly training simulator, in which the user could pass through objects with the virtual tool. However, in scientific visualization applications, the displayed torque could prevent the user from orienting the data probe to reduce visual occlusion, leading to a frustrating experience. Ideally, the probe should behave akin to the tool of an artist: it should not hinder the motion of the hand and support exploration in the best possible way.

Implementations of the traditional visualization pipeline typically provide a limited set of interactive data exploration capabilities. Tasks such as finding and measuring features in the data or investigating the relationship between variables of interest may be easier to perform with more natural data exploration tools. For example, the probe can be constrained to a selected isosurface of a scalar field, so the user can quickly explore

the corresponding vector field behavior by interactively seeding a set of streamlines near the surface. Although haptic interface research is still in an early phase, recent advances in the mechanical design of haptic interfaces allow for such sophisticated uses in graphics and visualization applications.

Haptic feedback has three related uses in scientific visualization applications. First, a rich lexicon of kinesthetic and tactile cues can be used to display the salient features in the data. For example, surfaces can be rendered with additional information overlaid to indicate discontinuities, transitions, and inhomogeneous regions [158, 114]. Haptically guided data exploration can aid the user in various tasks, such as finding and measuring features or investigating the relationship between several quantities in multifield data sets [77, 121]. Second, force feedback can improve user performance in data manipulation tasks, such as molecular docking [23, 115] or interactive segmentation [173]. Third, the associated user interface can benefit from haptic assistance. For example, constraining forces can be added to guide the user during selection and manipulation of interface elements or to indicate feasible configurations of a selected object [124, 134, 131, 95].

1.4 Thesis Objectives

This dissertation develops interactive global and local visualization techniques for the exploration of volumetric data sets in three-dimensional haptic virtual environments. The primary goal of the work is to develop techniques that augment the interface with suitable visual and haptic interaction methods for guiding the user in the data. To provide maximum flexibility to the user during exploration, the proposed visual and haptic rendering techniques do not require the extraction of intermediate geometric representations from the data. The techniques are illustrated with examples for representative visualization applications. The effectiveness of the approach is evaluated via a formal experiment and comparison with other existing approaches.

The success of the work is measured at multiple levels. For the rendering algorithms developed to maintain the required visual update rate for immersive applications, performance measurements and comparisons with standard techniques are provided. The calibration and registration algorithms are evaluated using validation measurements. The effectiveness of the combined visual and haptic interaction methods are assessed subjectively by expert users and quantified objectively in a controlled experiment.

1.5 Document Organization

The dissertation is organized as follows. Chapter 2 presents background information and summarizes related work on the components of the dissertation. Chapter 3 provides a brief overview of the testbed virtual environment [20, 75] used for the development of the data exploration techniques. The calibration methods [76, 79] developed for the coregisteration of the system components are described in Chapter 4. Strategies for interactive global and local volume visualization are given in Chapter 5. A general point-based haptic interaction method based on volumetric constraints [77] is presented in Chapter 6. The haptic rendering method is evaluated in Chapter 7 as part of a controlled experiment conducted to quantify the contibution of haptic guidance to vector field exploration tasks [78]. Example applications of combined visual and haptic exploration modes are presented in Chapter 8. Chapter 9 concludes the dissertation and discusses possibilities for future research.

CHAPTER 2

BACKGROUND

The goal of the dissertation is to develop interactive global and local visualization techniques that are suitable for haptic immersive environments and effectively combine flexibility with ease of use for typical visualization tasks. The following sections summarize research results from the visualization, immersive environments, and haptic rendering literature that are relevant to the methodology and specific techniques developed in subsequent chapters of the dissertation.

2.1 Interactive Data Exploration

Scientific visualization is the application of computer graphics techniques to aid in the understanding of complex numerical representations of scientific concepts or results. Traditionally, the role of scientific visualization is to create illustrations of a particular scientific phenomenon represented by the data. Due to the complexity of the phenomena and the large number of parameters that control the visualization, effective illustrations are often created via an iterative data exploration process. The level of interactivity provided by the visualization system is important during data exploration, because it is correlated to the effort required to produce an outcome that effectively conveys the intended information. The need for interactive data exploration is further justified by the fact that for complex volumetric data sets there is no single image that can illustrate all aspects of the data.

2.1.1 Time-Critical Volume Rendering

Even though volume rendering is arguably the most flexible and expressive global visualization technique, it has found limited use in immersive data exploration applications [156, 17]. One reason is the associated high rendering cost. Today even the fastest hardware-accelerated techniques are not able to achieve real-time updates without sacrificing overall image quality. To circumvent the fill-rate limitation, typical implementations

reduce the sampling rate and the viewport size during user input. Reducing the sampling rate, however, hides details in the volume and is not suitable for interactive exploration tasks, such as volume clipping or data probing. Advanced algorithms that use complex fragment shaders or multiple buffers can further increase the burden on the fragment processors and the texture and framebuffer memory caches [93, 80]. In addition, to avoid sampling and quantization artifacts, it is necessary to sample the volume in fine steps and use high-precision buffers for compositing [145]. For these reasons, high-quality real-time full-scale volume rendering has only been possible to achieve for very small data sets, limiting its use in practical applications.

Preintegrated classification is an effective method for reducing the sampling rate without aliasing artifacts [44]. The main idea behind preintegrated volume rendering is to assign colors and opacities to ray segments instead of sampling locations. Assuming that data values vary linearly along a segment, it is possible to precompute the corresponding color and opacity for all combinations of data values and store them in a lookup table. The preintegrated values are computed with high accuracy and are obtained from the table during rendering. Thus, the costly sampling and compositing steps along the ray segment are replaced with a simple table lookup. The linear variation of the data values is a reasonable assumption for small integration steps. However, the method can only be used with one-dimensional transfer functions or special classes of multidimensional transfer functions [93]. Furthermore, updating the lookup table during transfer function manipulation is an expensive operation. Section 5.5 describes a method that moves the precomputation step entirely to the graphics card.

Multiresolution volume rendering techniques build a hierarchical data structure to decompose the original dataset into multiple levels of detail (LOD). Typically, a partially populated octree of texture bricks is used for hardware-accelerated implementations [104, 182]. The goal is to speed up rendering by fitting the required bricks into texture memory and adaptively sampling the data according to the selected levels of detail. Several factors can be taken into account for LOD selection, including viewing distance, projected screen area, relative field-of-view, average opacity contribution, and region of interest [104, 182, 108]. Modifications to the standard texture-based volume rendering technique are needed to reduce visual artifacts at the boundaries between bricks from different levels [182]. First, to ensure interpolation continuity, data values from coarser level bricks are copied into finer level bricks at the boundaries. Second, the opacity transfer function α for

sampling distance d is adjusted from the reference opacity function α_0 and sampling distance d_0 according to the following formula.

$$\alpha = 1 - (1 - \alpha_0)^{\frac{d}{d_0}} \tag{2.1}$$

Third, spherical shell sampling [104] or clipping techniques [182, 183] are used to ensure that switching between two different sampling frequencies does not yield compositing artifacts. Spherical slicing is used to guarantee that sampling transitions happen in screen space, which works well for monoscopic displays. Clipping methods allow for arbitrary changes in the sampling frequency throughout the volume. Section 5.4 presents an extension of the volume clipping technique that accomodates preintegrated classification.

There are two problems with using the fixed octree-based multiresolution hierarchy for interactive data exploration. First, interpolation continuity does not guarantee smooth transitions across the brick boundaries. In addition, filtering and subsampling changes the topology of features in the data. Depending on the frequency contents of the data and the transfer function, rapid changes in value across the boundary frequently results in rapid changes in topology, which can cause disturbing visual artifacts. Second, the fixed structure of the octree provides only limited support for interactive specification of a region of interest (ROI) in the data. Popping and flickering artifacts often result when switching between levels as the ROI moves from one brick to another [108].

Recently, it has become possible to incorporate the classical acceleration techniques of early ray termination and empty space skipping into hardware-assisted volume rendering implementations [99]. Both techniques rely on the early depth test, which is also the basis for the techniques presented in Chapter 5. Early ray termination requires the periodic examination of the already accumulated opacities in the framebuffer. For pixels with the opacity above a given limit, the corresponding depth value is set such that any further operations on the pixel are skipped. Empty space optimization uses a small tag volume for marking empty blocks in the data. The tag volume is periodically examined and sample pixels in empty blocks are marked inactive using the depth buffer. Neither of the two techniques affects the quality of rendering, but the degree of speedup depends on the selected transfer function. It has been shown that the worst case scenario can incur a 30% performance overhead [99].

Clipping techniques can be used to render only a subset of the data for interactively probing large volumes [9] and to increase rendering speed in immersive volume rendering applications [21, 17]. These approaches, however, do not allow the specification of

arbitrary clip regions and provide only limited contextual information about which subset of the data is displayed to the user. Recently, it has become possible to use programmable hardware for efficient volume clipping by arbitrary convex objects [183]. Volume clipping allows the user to interactively select and explore regions of interest in the data set. A similar depth-based approach is used for the implementation of interactive lenses described in Section 5.3.

2.1.2 Interactive Lenses

Detail-in-context viewing is an important user interface component of systems developed for interactive visual inspection, exploration, and manipulation of complex abstract and 2D image data. Examples include the TreeJuxtaposer system developed for the comparison of information trees comprising several hundred thousand nodes [125] and the commercially available pliable display tools for detail-in-context viewing of multiple image attributes [73].

Interactive lenses were introduced to computer graphics as part of see-through interface [10]. Image space magic lenses appear on a virtual transparent sheet between the application and the mouse cursor and are manipulated in a manner that has a natural and intuitive analogue. Magic lenses can be used to perform a diverse variety of tasks, such as in-place editing and manipulation of images, modifying the visual appearance of application objects, enhancing data of interest, suppressing distracting information, and showing multiple data attributes. Later work presented a generalized formulation for the detail in context problem and developed several techniques for multilevel magnification of two-dimensional information [86].

In volume visualization, interactive lenses were first used for multiresolution reconstruction and time-critical rendering of isosurfaces. The MagicSphere [29] is a three-dimensional widget that defines a spherical region of interest in the data and classifies the extracted isosurface elements into internal, external, and border elements. A variety of visual effects can be achieved by applying different viewing filters to the internal and external elements. The border region was used for transitioning between the internal and external elements via alpha blending, which can result in visual artifacts for certain combinations of filters. The technique described in Section 5.7.3 uses a similar fragment classification scheme for multiresolution focus and context volume rendering. However,

the method yields smooth transitions by blending the data values instead of the classified colors and opacities within the border region.

Object space lenses proved useful for reducing clutter and occlusion in an immersive flow visualization application [52]. Volumetric lenses overcome the viewpoint dependency of see-through lenses and are more suitable for data exploration in immersive and collaborative environments. The confined volume of the lens also allows focussing to an explicitly specified spatial region as opposed to the implicit volume defined by the view-space lens. In this work the backfacing sides of the box were used to reduce clutter by culling visualizatoin primitives behind the focus, which also facilitated the separation of primitives inside and outside the box via a two-pass rendering technique and a combination of the depth test and the stencil test.

Magic lenses have also been proposed for enhancing the display of 3D polygonal models. Initially, only box lenses could be implemented due to the lack of hardware support for programmable per-fragment clipping [174]. Thus, clipping planes were used instead and the scene had to be rendered in seven passes. With the appearance of programmable graphics hardware, it has become possible to perform multiple depth tests in the fragment pipeline. A second depth test allows the use of arbitrary convex lens shapes [147]. By classifying objects into three regions based on their position with respect to the lens, it is possible to visualize transparent scenes and the number of passes is considerably reduced. VTK provides an example of how to use cutting spheres and planes for creating a combined view of features [155]. It is also possible to specify a region of interest with a high-dimensional transfer function [170] or from the output of interactive segmentation [107].

In early work, a focal region guided volume rendering technique was proposed that combines two different rendering modes to show continuous detail in the region of interest and to reduce occlusion by the surrounding context [189]. A similar idea is used in two-level volume rendering, where different transfer functions, and rendering and compositing modes are integrated to create a more informative display of segmented objects in medical data sets [59]. A 3D X-ray lens was used to combine volume rendering and icon visualization in a stereoscopic system for exploring the correlation between two scalar fields [163].

Distortion lenses were recently incorporated into hardware-assisted volume rendering implementations [103, 30, 178]. Magnifying lenses facilitate the exploration of a specific

region of interest without losing the surrounding context, which otherwise happens when zooming in the data. One approach uses a modified ray casting algorithm to accomodate screen space lenses for generating magnified or fish eye views [178]. A transition region is added between the lens and the context to provide continuous observation of the objects of interest during manipulation of the lens. The magnification lens described in Section 5.7.1 incorporates a similar transition region, but uses a more efficient slice-based volume rendering implementation.

The VolumeShop system uses a selection volume for specifying a particular structure of interest in the data [24]. The system provides different transfer functions for the focus and the context to create cutaway and ghost views of a selected object. The implementation exploits the early depth test to render and composite the focus and the context in separate passes, similar to the method presented in Section 5.3.

ClearView is an intuitive focus and context volume visualization tool that was developed based on the principles of technical illustration techniques [98]. ClearView assumes that the focus is represented by a single isosurface layer and that the context can be decomposed into several such layers. Position, normal and curvature information is extracted for each layer during the first stage of the rendering algorithm. In the second stage, defered shading techniques are applied to each layer to enhance or supress features in the data. To facilitate understanding of the spatial relationship between the focus and the context, ClearView uses importance measures to fade out structures of the context in proximity of the focus. The curvature based importance measure ensures that characteristic structures of the context are shown even in proximity of the focus. The distance based importance measures allow selective visualization of structures close to the context layers or to the viewer.

Interactive lenses can be classified according to the shape of the focus region, the type of visualization and interaction methods used in the focus and the context, and the interaction space for the lens [30]. Chapter 5 follows a similar classification and implements lenses from various categories for volume data exploration tasks.

2.2 Immersive Environments

Immersive environments are particularly suitable for volumetric data exploration applications. Immersive displays provide additional perceptual cues for exploring and manipulating three-dimensional objects. For example, head-tracked stereo displays combine binocular perspective, motion parallax, and vestibular cues to convey information

about the location and shape of the objects. Systems with a wide field of view can use the peripheral visual field and optic flow for more effective navigation in the data. Hand tracking provides dynamic occlusion and proprioceptive cues to aid 3D selection and manipulation tasks.

Unfortunately, the effective use of the comprehensive perceptual feedback provided by immersive environments is currently hindered by engineering limitations. For example, a head-tracked stereo display with a tracked hand-held interaction device may not offer any combined advantages if the resulting perceptual cues conflict with each other. Examples of the hardware limitations include inaccurate registration of the system components, low resolution displays, and high latency input and output devices. Examples of software limitations include lack of time-critical rendering methods for maintaining the required visual update rates and lack of support for effective interaction techniques.

The following section provides an overview of calibration and registration methods for combined visual and haptic displays. Calibration techniques for position tracking devices are discussed in the context of electromagnetic tracking systems. The subsequent section summarizes how interaction techniques have been used for immersive data exploration applications.

2.2.1 Calibration and Registration

Many immersive virtual reality applications benefit from precisely calibrated system components. Greater accuracy is desired to provide users with a more compelling experience, and to increase precision and reduce frustration during six degree-of-freedom interaction tasks. Experimental testbed systems require very accurate registration, otherwise registration artifacts would be difficult to separate from other factors [167]. Visual artifacts caused by registration errors include misalignment of real and virtual objects, and the notorious "swimming" effect, which is the motion and changing shape of stationary objects as the user moves around in the environment. In addition, it is important to match the generated visual and haptic cues by colocating the workspaces of immersive visual and haptic displays.

There are a number of factors affecting the overall accuracy of an immersive display system [143, 38, 66, 146, 69]. Registration error sources can be categorized according to whether they produce geometric or optical distortions. Geometric errors result from inaccurate tracking, system delay, misalignment of coordinate systems, and imprecise viewing and interaction parameters. Optical errors, caused by the limitations of the

image generation subsystem, are manifested by convergence and aliasing problems, display nonlinearities, and color aberration. Haptic rendering fidelity largely depends on the structural and dynamic characteristics of the haptic interface, the accuracy of its kinematic description, the update rate, and the control algorithm used to produce reaction forces and torques.

There are several reasons why accurate registration is necessary in immersive environments. One of the artifacts caused by misalignments occurs in multiscreen projection environments, where visual errors create a discontinuity in the shape of objects displayed across multiple screens. This is not an issue for single-screen displays. However, as demonstrated in Section 4.5, head-tracking errors cause the perceived virtual space to warp. As long as objects are kept small and close to the projection surface, the visual distortion is not significant. For large objects, improper scaling of the visual space caused by viewing errors yields incorrect perspective, because the back parts of objects will appear larger or smaller than desired. Also, if objects are located further away from the screen, either in front or behind, the shearing of the visual space will cause noticable swimming. Finally, registration errors cause the visual representation of an interaction device to move relative to the device itself. This is a serious problem for back-projected displays, because users can see both the physical and virtual props at the same time.

2.2.1.1 Calibration of Magnetic Tracking Devices

Electromagnetic tracking systems have been used extensively in virtual reality research and applications. Magnetic trackers determine the position and orientation of receivers relative to a transmitter by generating and measuring orthogonal electromagnetic fields [140, 100]. These systems are preferable to other tracking technologies because they are relatively inexpensive, cover a reasonably large workspace, provide fairly good resolution with acceptable jitter, are convenient to use, and do not suffer from line of sight problems [41]. However, their accuracy is seriously degraded by magnetic field distortions due to metal and electronic equipment located in the environment. The error between the actual and reported position and orientation of the receiver increases in proportion to its distance from the transmitter [128]. In fact, position errors can be as high as several feet which limits the usable range of these devices [54].

In the past decade, researchers have proposed a number of methods to improve magnetic tracking accuracy by measuring and compensating for the field distortions.

Calibration is possible provided that the magnetic excitation vectors produced by the transmitter remain linearly independent within the working range of the device [140]. A crucial component of tracker calibration is the method of deriving error corrections from the experimental data. Typically, calibration methods implement one or more error correction techniques from the following categories:

- **Analytical**: Analytical techniques assume that the distortion is a function of the receiver position that can be closely approximated by a higher order polynomial. This approach was originally proposed by [140] and later evaluated in [25, 91]. Since it has been shown that tracker noise is proportional to the fourth power of the transmitter-receiver separation distance [128], the use of approximating polynomials of the same degree is physically appropriate and has proven to work reasonably well.

- **Global interpolation**: Instead of fitting an analytical model to the collected data, global methods apply scattered data interpolation techniques to describe the distortion field explicitly [187] or to construct a lookup table (LUT) for local interpolation [36].

- **Local interpolation**: Local methods are based on a uniformly spaced LUT from which the error correction is calculated using trilinear interpolation. A uniform calibration table can be built in the distorted tracker space from a systematic data collection procedure [54] or by resampling irregular measurements onto a rectilinear grid [110, 36, 22]. A more direct approach is to construct the table in the actual workspace [25, 92], in which case fast interpolation and grid traversal algorithms are required to calculate the correction values at runtime. The efficacy of these methods depends on the granularity of the table, because the distortion is assumed to be linear within a grid cell. Comparative studies have shown that local methods perform at least as well as analytical approaches, but their complexity is higher [92].

Data collection techniques developed in the past differ in achievable accuracy and ease of use. Direct methods are based on an external metrology system registered to the transmitter to determine the exact location of the receiver [54, 110]. Indirect approaches rely on human perception to approximately align the experimental measurement apparatus, which is typically a platform [187, 91], pegboard [25], or jig [22]. Such methods are generally unsuitable for measuring orientation, but it is possible to determine orientation

distortion from a set of constrained poses [91]. Approximate corrections can also be obtained by superimposing real and virtual targets located in the workspace [36].

In the past, the majority of research in this area focused on correcting the position component of the error. However, it has been noted that orientation correction is necessary for hand tracking to reduce user confusion and frustration, especially for manipulation tasks [187]. Initial research to correct orientation errors concluded that the orientation error is a function of the receiver position and orientation, thus any correction technique is impractical because of the large number of data points required [110]. The results in Section 4.2.5 indicate that orientation correction is possible with a level of accuracy comparable to that obtained by position correction techniques.

2.2.1.2 Registration Techniques for Immersive Displays

The majority of previous work has concentrated on registration issues for head mounted displays and desktop augmented reality systems [143, 55, 7, 146, 69]. A careful examination of factors influencing the accuracy of a desktop head-tracked stereo display illustrates the complexity of the issues [38].

A registration procedure utilizing a precision surveying theodolite was developed for colocating the visual and haptic workspaces of the nanoWorkbench [56]. The authors mentioned that a better approach would be to develop a semiautomatic calibration method by attaching a rigid extension to the end-effector of the haptic interface, similar to the apparatus described in Section 4.4. A calibration methodology for augmented reality experimental testbeds was evaluated for the Virtual Hand Laboratory [167]. The Reachin API has a built-in calibration procedure that can be used to align a virtual tool representation with the PHANToM stylus [1]. Attempts have also been made to characterize and improve the positioning accuracy of the PHANToM haptic interface [142, 175, 28].

In early work, a method for pointer and object calibration was successfully used for a monitor-based augmented reality system [168]. More recently, a comprehensive registration procedure suitable for both head-mounted and head-tracked displays was proposed [53]. The method described in Chapter 4 adapts the techniques of hotspot, world, and viewpoint calibration to the display configuration described in Chapter 3 of the dissertation.

2.2.2 Immersive Data Exploration

The Virtual Windtunnel combined a head-tracked fully immersive visual display and a tracked glove controller to provide an intuitive interface for the exploration of unsteady fluid flow data [27, 123]. With this system, users were able to interactively create and observe particle traces by manipulating groups of seed points via simple hand gestures. The rapid visual feedback and the spatial input device provided an effective way to quickly sample the data without clutter. In addition, users could easily investigate nearby features by moving the seed points with the glove and generating the particle traces in real time. A similar pinchglove interface was used for data probing and object manipulation in a semi-immersive workbench environment [132].

A fully immersive CAVE application was developed for exploring numerical flow simulations through a model of a coronary artery graft [46]. The goal of the system was to allow the user to identify potential sources of future lesions by examining the effect of changes to the geometry and to the flow. Gestural navigation was reported intuitive and easy to learn, but tedious to use over long distances and not sufficiently accurate for some of the tasks.

Two-handed interaction techniques have proven useful for immersive scientific visualization applications. These methods exploit the assymetric division of labor between the hands. Typically, the subdominant hand is responsible for low-frequency contextual operations, while the dominant hand is used for fine-grain manipulation [35]. An example of this concept can be found in an application developed for the exploration of geoscientific data with an immersive workbench system [61]. In this application, the user could simultaneously specify an arbitrary slice of the data and inspect the details in the slice using a virtual flashlight attached to the input device in the dominant hand. Similarly, the Stereoscopic Field Analyzer used two identical hand-held tracking sensors with additional buttons mounted for command input [42]. The goal of the system was to provide a comprehensible display of flow data via stereoscopic glyph visualization and direct manipulation techniques. The subdominant hand was used for scene navigation, menu input, and bounding box axis selection, and the dominant hand performed data probing and selection tasks.

One problem with unconstrained two-handed spatial input is lack of support for the hands. Large-scale environments and applications that require an intuitive, precise, and comfortable workspace can benefit from the use of passive physical interface props [63].

A popular prop for two-handed interaction is the combination of a hand-held semitransparent panel and a pen [154, 33, 184, 37]. In addition to providing a way for embedding 2D controls in a 3D environment, the panel can further assist with typical visualization tasks, such as the placement of cutting planes and interactive lenses in the data. Support for the hands can also be provided by the display surfaces in semi-immersive workbench environments [48, 161].

Recently, a CAVE application was developed for the inspection of DT-MRI data sets [188]. Static geometric representations of relevant features were extracted in a preprocessing step. The resulting geometry was decimated and displayed together with anatomical landmarks and T2-weighted MR slices. The display of 2D slices complemented the static geometry, since users could examine the details and validate their assumptions about the model by sliding the slices back and forth. Feedback from the informal evaluation indicated that showing additional details would be a desired capability, especially around a selected region of interest.

The Crumbs application was the first example that combined the advantages of volume visualization and immersive environments to support the identification of structures in biological imaging data [18]. The application included an immersive curve editing tool that allowed users to mark and measure features in the data by placing control points for spline curves directly in 3D space. Volume rendering was used to guide the manual feature identification task. To achieve interactive frame rates, a multiresolution approach was proposed that combined rendering a subsampled version of the entire dataset with a subset of the data at the original resolution within a small window in space called the clear box. The position of the window remained stationary in space with an orientation orthogonal to the view direction, allowing the user to concentrate on the selected area of interest.

Volume rendering has proven useful for immersive modeling applications. For example, a dextrous environment was developed for manual segmentation and editing of vessel models from CT data [160]. Such modeling application could be used to incorporate patient-specific information into medical training simulators. The semitransparent volume display was primarily used for precision tuning of the vessel model. The human expert could easily use the system to decide what aspects of the data to ignore as opposed to automated segmentation and simplification algorithms. To facilitate smooth interaction, only the required subset of the volume was displayed to the user.

A region-based approach for interactive volume exploration and simulation was developed in the 3DIVE system [17]. The standard view-aligned slicing algorithm was modified to facilitate the display of multiple regions from the same volume. Each region could be transformed separately, so they appeared as distinct volumetric objects, even though the assigned data values originated from the same data set. Filtering could be performed by keeping a local copy of the data for each region. Finally, the slicing algorithm was adapted to accomodate volumetric collision detection between the region objects.

The VIRVO volume rendering library was created for the visualization of scalar data sets in projection based virtual environments [156]. The goal was to provide users with an intuitive interface for the manipulation of transfer functions with a 3D input device in a CAVE environment. The transfer function editor was placed on a floating menu in space in front of the user. Sliders were replaced with rotary knobs, which were found more convenient for 3D manipulation from a distance. In addition, knobs increased the accuracy of value adjustments, since their behavior was independent of the distance of the user from the interface panel. The user could also switch between an interactive and a high-quality rendering mode or select only a subset of the data for display. Even though users in general did not complain about rendering quality, the developers concluded that a level of detail rendering technique is required to improve interactivity of the system. Related work on immersive volume visualization found that traditional slider controls are more suitable for workbench environments [21] and for hand-held physical interaction panels [184].

2.3 Haptic Rendering

The following subsections review the evolution of point-based haptic rendering techniques with an emphasis on scientific visualization applications. The techniques are classified according to the data representation used into surface and volume methods. Within each category force field rendering techniques are compared with constraint-based approaches. The review also describes how the methods evolved from simple problem-specific approaches towards more general solutions.

2.3.1 Surface Rendering

The majority of haptic rendering algorithms are geometric in nature, since they deal with the problem of interacting with various boundary representations. Surface rendering requires a suitable geometric model, efficient algorithms for rapid collision

detection and minimum distance queries, and a model for generating contact forces from the probe-surface interaction. Collision detection algorithms exist for a variety of boundary representations, including polygonal, parametric, and implicit surfaces. These algorithms combine of global and local distance queries to track the geometry closest to the interaction point.

Contact forces are usually modeled by interactions between the probe point and a rigid viscoelastic surface. The earliest surface rendering method augmented objects with a repulsive force field that pushed the probe out as soon as it penetrated the boundary [117]. The approach worked well for simple and smooth shapes, but failed to reproduce the crisp force transitions at sharp edges, as illustrated in Figure 2.1. Thin structures were also missing, because as soon as the probe entered the object from one side, it was pushed out immediately through the opposing side. Surface constraints were introduced as a solution to the problems of the force field method [190, 149]. The interior force field was replaced with a force computed between the probe position and its ideal location on the boundary surface, called the proxy point or surface contact point. The force computation step was augmented with a surface tracing step that updated the location of the proxy based on its previous location, the displacement of the probe, and the surronding geometric configuration. Since incremental surface tracing finds the locally closest surface contact

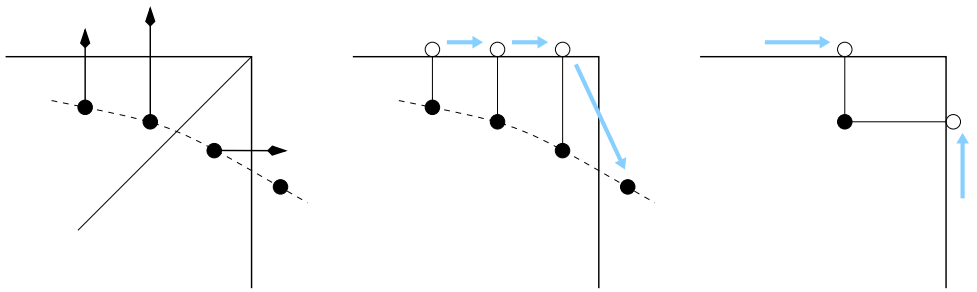


Figure 2.1. Evolution of haptic surface rendering methods. (a) The earliest surface rendering technique augmented objects with an internal force field that pushed the probe towards the surface of the object. The method fails to reproduce the sharp transitions at the edges, since the field pushes the probe towards an adjacent face as soon as it enters the corresponding volume. (b) A restoring force can be computed by tracing the locally closest surface point to the proxy. (c) Different forces can be obtained for the same probe position, depending on the path taken by the probe. The proxy can also be viewed as a memory element in the system that stores state information.

point to the probe, it implicitly takes into account not only the current probe location, but also the path already taken by the probe, as illustrated in Figure 2.1.

Typically, a virtual spring and damper combination is used to mechanically couple the probe with the proxy during contact. From a more general point of view, surfaces are represented by unilateral constraints that prevent the proxy from penetrating the object. Point-based surface rendering research has focused on improving the perceived crispness of surfaces and on augmenting the surfaces with various material properties to create realistic and convincing virtual objects [151, 118, 164].

2.3.2 Volume Rendering

Early work on using haptic feedback for scientific visualization developed and evaluated techniques that mapped scalar and vector data values directly to force and torque components [84]. Force fields were also used for guiding the user in a molecular docking application [23]. The majority of methods developed for haptic display of data properties can be described by a functional relationship between the reflected force and torque vectors, the probe state, and local data measures [112].

$$F = F\,(X,\, D,\, T) \tag{2.2}$$

In Equation 2.2, the feedback force and torque components are collected into F, X denotes the state, typically position $\mathbf{x}$ and velocity $\dot{\mathbf{x}}$ of the haptic probe, D represents a set of local data measures at the probe position, and T stands for a set of haptic transfer functions and rendering parameters. Popular force field rendering techniques include density modulated viscous drag for scalar data [5, 136] and direct display of vector data [84, 49, 120, 40].

$$\mathbf{f}(\{\dot{\mathbf{x}}\}, \{s(\mathbf{x})\}, \{k(s)\}) \;=\; -k(s(\mathbf{x}))\,\dot{\mathbf{x}} \tag{2.3}$$

$$\mathbf{f}(\{\}, \{\mathbf{v}(\mathbf{x})\}, \{k\}) \;=\; k\;\mathbf{v}(\mathbf{x}) \tag{2.4}$$

The feedback gain k is adjusted according to the scale and magnitude of the data measures and the capabilities of the haptic interface. Equation 2.3 modulates viscous resistance according to the data value at the probe and Equation 2.4 computes a force directly proportional to the local field vector. More complicated examples include the vortex core identification mode [105] and the transverse damping mode [136], developed specifically for the exploration of vector field data.

$$\mathbf{f}(\{\}, \{\mathbf{v}(\mathbf{x}), \boldsymbol{\omega}(\mathbf{x})\}, \{k\}) \;=\; k\,(\boldsymbol{\omega}(\mathbf{x}) \times \mathbf{v}(\mathbf{x})) \tag{2.5}$$

$$\mathbf{f}(\{\dot{\mathbf{x}}\},\{\mathbf{v}(\mathbf{x})\},\{k\}) \quad = \quad k\left(\dot{\mathbf{x}} - \frac{(\dot{\mathbf{x}}\cdot\mathbf{v})\,\mathbf{v}}{\|\mathbf{v}\|^2}\right) \tag{2.6}$$

In both equations $\boldsymbol{\omega}(\mathbf{x})$ denotes the vorticity of the field. Equation 2.5 computes a force that attempts to push the probe towards vortex cores in the data. For example, if the flow is locally helical, the force vector points towards the center of the local helix [105]. Equation 2.6 computes a viscous drag force transverse to the flow, which opposes rapid user motion perpendicular to the local orientation of the field.

Even though force field rendering techniques map data properties directly to force and torque output in an intuitive and straightforward manner, they suffer from the following two limitations:

- **Limited expressive power.** In the real world, the haptic sense is primarily used for extracting physical object properties. Determining the shape, orientation, material stiffness, texture, and inertia of an object is often needed for manipulation or for augmenting the available visual information. Some of the physical object properties are well suited for the haptic presentation of features in scientific data. However, it is difficult to display volumetric features in a purely functional form. Even though it is possible to create static force fields that represent sharp force transitions or constraining directions, it is not always obvious how to construct these fields for a given dataset. In addition, the spatial resolution of force fields is limited, and it is often unclear how to combine multiple fields without ambiguities. Perhaps the most serious drawback is that small features cannot be captured adequately by force fields. The reason for this is the same as for surface rendering: the notion of memory is missing from the functional formulation [190, 150, 112]. For example, using force fields for displaying isosurfaces yields less informative feedback than traditional surface rendering approaches [5, 82, 105]. Also, mapping a vector field directly to a force field provides a poor indication of the local field direction.

- **Parameter tuning.** Force field rendering techniques implicitly capture the device capabilities in the rendering parameters. Applying a force as a function of the probe state only can easily result in instability, especially when several rendering modes are combined. It is a very difficult and tedious task to tune the dynamics of the sampled-data system comprising the computed feedback force, the haptic interface, and the human operator by searching for an appropriate set of rendering parameters.

Similarly to surface constraints, volumetric constraints provide a solution to these problems. Even though constraining forces can be constructed directly from force fields, a larger variety of effects are possible when a proxy point is incorporated into the rendering algorithm. As discussed in Section 2.3.4, the proxy also acts as a stabilizing virtual coupling and represents the ideal location of the probe in the data.

2.3.3 Haptic Constraints

Haptic constraints have proven useful in several telerobotics and virtual environment applications. In early work, synthetic haptic overlays called virtual fixtures were shown to improve operator performance in robot teleoperation tasks [148, 153]. In the same way as their physical analogues, virtual fixtures improve the precision and speed of manipulation in addition to reducing the mental and physical workload required to perform the task. An important consequence of using virtual fixtures is that the added sensory feedback alters the conceptualization of the task, such that a successful execution not only looks correct, but also feels correct [148].

Constraints provide a general foundation for developing haptic rendering algorithms for scientific data sets [77]. The first application of haptic constraints to scientific visualization was a 2D graphing system developed specifically for the visually impared [50, 49]. In this system, 2D line plots were augmented with a static force field that attracted the cursor towards the line segments. The problem with this approach, and with force fields in general as discussed in Section 2.3.2, is that the field created for one segment may conflict with the field for a neighboring segment, resulting in ambiguous feedback when transitioning between the segments. To circumvent this problem, only one segment was made active at a time based on the location of the cursor.

Static force fields have also been used for constraining the haptic probe during the exploration of 3D segmented voxel data [8]. Two force fields were combined in a 3D vessel tracing application. Each field was derived from the gradient of a scalar distance field. The first field was defined as the Eucledian distance of a point within the interior of the object and the segmented boundary and was used to push the probe towards the center of the object. The second distance field was defined as the distance from a given starting point and directed the cursor from the starting position towards a goal position.

A 2D haptic device was used to provide depth cues in a graphical application developed for the visualization of volume angiograms [186]. In the vessel tracing mode, the probe was visually constrained to a selected vessel branch. The feedback force was proportional

to the depth gradient of the vessel centerline curve and pointed along the projected tangent direction. Thus, the user experienced stronger forces along the curve when the corresponding change in depth was larger. Evaluation of the method indicated that the technique was an efficient indicator of depth changes along the branches of the vessel network.

A variety of techniques have been developed for exploring isosurfaces in scalar density data. Similarly to graphical visualization, the normalized gradient vector can be used as the normal vector for the isosurface. In early work, force fields were created around isosurfaces by approximating the penetration of the probe into the surface with the difference between the scalar data value at the probe and the selected isovalue [5]:

$$d(\mathbf{x}_n) = \ c(s(\mathbf{x}_n) - s_0) \tag{2.7}$$

The penetration distance can also be approximated by the distance traveled by the probe along the gradient direction within the object represented by the isosurface [105]:

$$d(\mathbf{x}_n) = \sum_{i=1}^{n} \mathbf{n}_i \cdot (\mathbf{x}_i - \mathbf{x}_{i-1}) \tag{2.8}$$

The problem with the above approaches is that the given approximations are valid for small penetration distances and smoothly varying fields only. With either method, the surface contact point corresponding to a given normal direction and penetration distance is not likely to lie exactly on the isosurface. As the user moves the probe along the surface, the error adds up and the approximated penetration distance may be significantly different from the actual one. Thus, the resulting forces do not effectively capture small surface variations, and the displayed surface may feel different than what the visual representation indicates.

Rendering solid objects represented by implicit surfaces is a closely related problem. The first haptic tracing algorithm for implicit surfaces used an iterative solver to constrain the proxy exactly on the zero set surface [150]. The input to the solver is the projection of the probe to the tangent plane at the proxy. The iterative refinement is needed, because the numerical error from the first-order approximation accumulates quickly and the proxy can easily leave the surface during exploration. The correction factor is ensures that the proxy remains on the zero set surface after the update step. The algorithm described in Section 6.6 uses a similar technique for constraining the proxy to an isosurface. In related work, a slightly different version of the iterative refinement method based on binary search was proposed [88].

A variant of the iterative approach was developed for exploring isosurfaces in medical imaging data [12]. The algorithm is based on repeatedly moving the proxy along the local tangent plane until the direction of the surface normal and the difference vector between the probe and the proxy locations are parallel. A major problem with the approach is that it does not guarantee that the proxy remains on the surface after the update step. In addition, the method yields undesirable oscillations when the filtering kernels for the data value and the kernels for the gradient do not match.

A haptic rendering framework was developed for exploring isosurfaces in volumetric density data [112, 114]. In this work, three rendering modes are combined to provide natural haptic feedback to the user. The first mode constrains the proxy to a local isosurface patch by incrementally moving it along the tangent plane of the surface. By decomposing the motion of the proxy along normal and tangential directions, the reaction force can be constructed from independent components using haptic transfer functions. The second rendering mode creates friction along the isosurface by moving the proxy only when the distance between the probe and the proxy is above a given threshold. The third mode is added to display isotropic viscous resistance in areas where the gradient of the field is ill-defined. A similar isotropic drag mode was used in a geoscientific data visualization application [6]. The haptic rendering technique described in Chapter 6 is the generalization of the scalar data rendering methods above to vector, tensor, and multifield data sets.

Dynamic haptic constraints were successfully used in an application for browsing and editing families of animations [39]. The animations were mapped into the configuration space of the haptic device, such that each trajectory was represented by a control path of the probe. Two editing modes were implemented for the application. In the follow mode the user could replay animations and slightly change the visual output by pulling the cursor away from the control path. The strechy tubes mode allowed the user to modify an existing trajectory or to create a path for a new animation.

A number of passive and active interaction modes were developed for exploring scalar and vector fields in a CFD visualization application [172]. Passive modes automatically moved the probe along a selected trajectory in the data and the user was only an observer of the resulting visual and haptic feedback. In contrast, the active modes allowed the user to directly place the probe in the data. One of the active modes constructed constraining forces from a number of precomputed streamlines curves in the vector field.

The evaluation of the techniques indicated that users preferred to actively participate in the exploration process instead of passively observing it.

To provide support for the rapid development of haptics applications, software components for haptic constraints were incorporated into the MAGMA scenegraph API in the form of a haptic constraints library [71, 72]. The library included a class hierarchy for point, line, and surface constraints and also provides a mechanism for combining multiple constraints. In the constraint pipe algorithm, constraints are applied repeatedly until all of them are satisfied or the maximum number of iterations is exceeded. Since sometimes it is impossible to satisfy multiple constraints with a single proxy point, the output of the algorithm is not always well-defined.

The OpenHaptics Toolkit uses the OpenGL feedback buffer to capture geometric point, line, and polygon primitives for haptic rendering [83]. The captured polygon primitives can be used in unilateral or bilateral constraint mode. When multiple overlapping constraints are specified, the API will constrain the proxy according to decreasing order of dimensionality. This allows the user for example to snap the probe to a lower dimensional constraint embedded into a higher dimensional constraint. For example, the probe could be attracted to a point and constrained to a plane simultaneously.

Recently, a force-based approach was developed for combining multiple constraints represented by linear haptic primitives [113]. At every iteration the position of the proxy is updated by balancing the force from the probe with the forces from the primitives. The primitives are placed automatically according to the local properties of the data and the selected haptic mode. Since the proxy obtained from the force balance equation may not satisfy any of the constraints, it is not possible to implement precise directional haptic modes using this approach.

Volumetric constraints have also been proposed as an alternative to existing surface rendering algorithms. Examples include a 6DOF haptic rendering framework with algebraic constraints as the foundation [70] and a method for multicontact object manipulation [65]. The former approach is work in progress. The latter formulation simplifies the collision detection process, but requires a tetrahedral discretization of the workspace, which is difficult to obtain for large and dynamically changing scenes. Thus, it seems unlikely that these techniques will become a suitable replacement to surface rendering algorithms in the future.

2.3.4 Performance Considerations

A haptic interface is a programmable force-feedback device that generates mechanical impedances [31]. A mechanical impedance represents the dynamic relationship between the state of the probe and the reflected force vector. Ideally, the device should be capable of exhibiting a wide variety of impedances, ranging from perceptually zero impedance corresponding to unobstructed motion of the probe to perceptually infinite impedance corresponding to rigid constraints [136]. The performance of a haptic interface can be characterized in terms of its transparency, i.e., the ability to display the programmed mechanical impedances [32]. There is no formal definition for transparency other than it expresses the difference between the programmed and displayed mechanical behaviors.

An important requirement from any force-feedback system is that it should not exhibit instability in the form of unwanted oscillations. Unstable behavior is caused by the excess energy accumulated in the coupled system comprising the virtual environment, the haptic interface, and the human operator. In practice, stability is influenced by a number of factors, including sensor noise, sample and hold effects, numerical simulation errors, and inherent device impedance. One way to derive stability conditions for a given system is to represent the user and the virtual environment as passive linear two-ports connected with a virtual coupling network, as shown in Figure 2.2. The coupler is used to adjust the transparency of the bidirectional energy flow between the user and the virtual environment. In addition, the coupler acts as a low-pass filter between the haptic device and the virtual environment, limiting the maximum impedance that needs to be exhibited by the device and preventing the accumulation of energy in the coupled system [32]. The advantage of the two-port network model is that both impedance and admittance displays and environments can be considered for the design of the coupler.

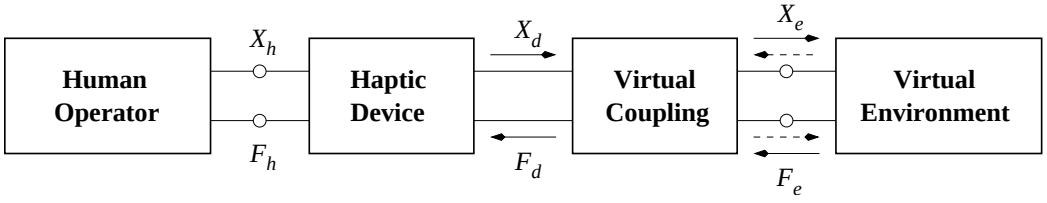


Figure 2.2. Two-port network model of haptic rendering. Impendance environments read the state of the virtual coupling element and return the corresponding force vector. Admittance environments act in the opposite direction as indicated by the dashed arrows.

Force field rendering techniques represented by Equation 2.2 correspond to using a tight coupling between an impendance environment and the haptic device.

$$X_e = X_d \tag{2.9}$$

$$F_d = F_e(X_e) \tag{2.10}$$

Equations 2.9 and 2.10 express the fact that the probe point and the proxy point are exactly the same. The result of the tight coupling is that it becomes impossible to ensure stability of the system even when both the human operator and the virtual environment are passive and stable. Since the biomechanical impedance of the human operator is difficult to predict, it is the task of the virtual environment designer to ensure system stability, typically via a tedious parameter tuning procedure [2].

In contrast, separating the probe point from the proxy allows for explicit control of the displayed environment impedance.

$$X_e = X_e(F_e) \tag{2.11}$$

$$F_d = -F_e(X_d, X_e) \tag{2.12}$$

Equation 2.11 describes the proxy update step from the coupling force for an admittance environment. The state of the proxy is updated such that the constraints imposed by the environment are maintained and the coupling force F_e is locally minimized. Chapter 6, Section 6.7 describes a proxy update strategy that supports multiple nonlinear constraints in the environment. The displayed force is the opposite of the coupling force, which depends only on the state of the probe and the proxy point, as indicated by Equation 2.12. Typically, a spring-damper coupling is used for point-based haptic rendering applications.

CHAPTER 3

THE VISUAL HAPTIC WORKBENCH

The Visual Haptic Workbench (VHW) is a testbed system developed primarily for haptic immersive scientific visualization [20]. In this chapter, issues related to the design, configuration, and general use of the system, including user interaction techniques and software development support, are discussed. The registration and rendering techniques described in subsequent chapters can be used to address some of the issues raised in this chapter.

3.1 Components and Design Considerations

The Visual Haptic Workbench is composed of a SensAble PHANToM 3.0L mounted on top of a Fakespace Immersive Workbench in an inverted overhead configuration, as shown in Figure 3.1. Head, hand, and stylus pose measurements are provided by a Polhemus Fastrak magnetic position tracker. Stereo images are generated by an Electrohome Marquee 9500LC projector and are reflected via folded optics onto the back of the nonlinear diffusion surface of the workbench. A pair of Stereographics CrystalEyes LCD shutter glasses, strobed at a 120Hz refresh rate, is used for stereo viewing. Typically, the dominant hand of the user manipulates the PHANToM stylus to experience haptic feedback from the virtual scene and the subdominant hand is used for system control tasks such as navigating a menu interface or manipulating a widget. A pair of Fakespace Pinch Gloves and a pair of 5DT Data Gloves are also provided for implementing gesture-based interaction techniques. Several custom modifications were made to the workbench hardware, including a "step to operate" footswitch instead of the original "push to interrupt" switch, provided as a more convenient safety mechanism, a registration apparatus for placing the PHANToM in a fixed position on the surface of the workbench during encoder initialization, and an inexpensive 6DOF interaction device, the I^3Stick [19]. The system is constructed such that it can be connected to a desktop PC with the latest available graphics accelerator without further modifications.

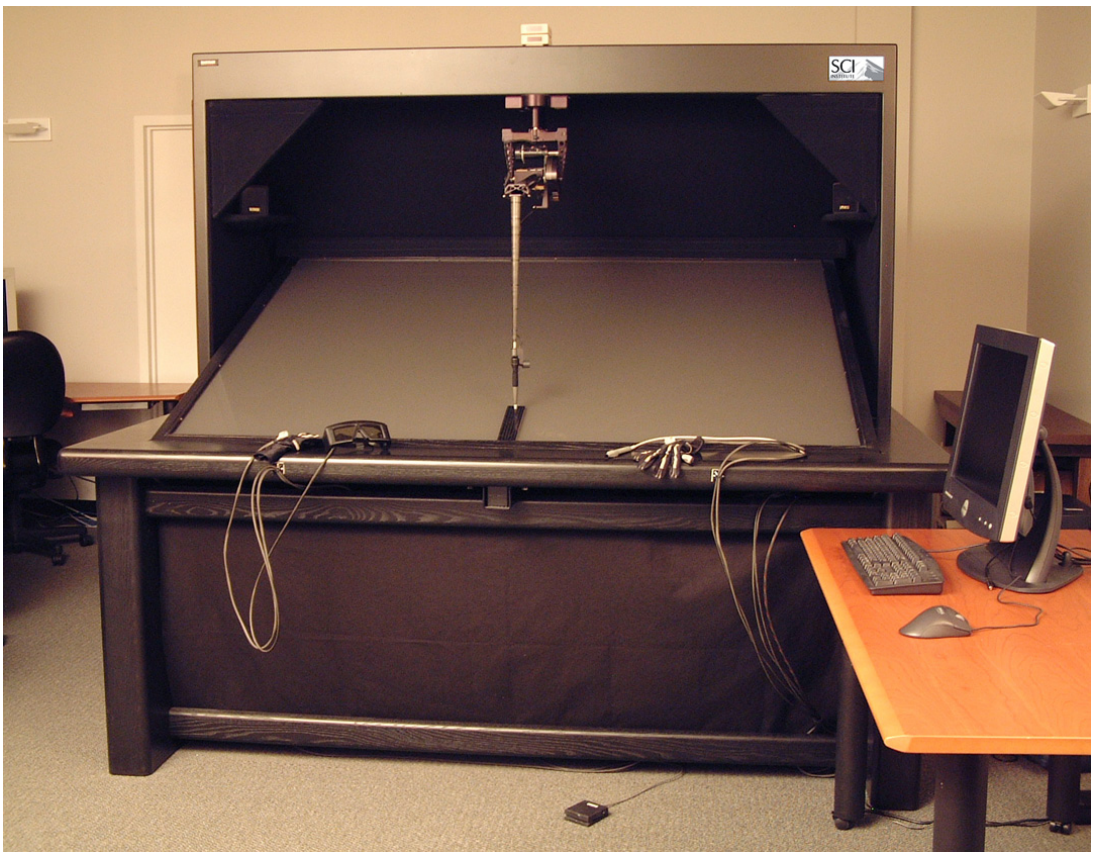


Figure 3.1. The Visual Haptic Workbench. The Visual Haptic Workbench integrates a large workspace SensAble PHANToM with a Fakespace Immersive Workbench.

Several factors influenced the selection and placement of the system components. Compared to other similar systems, including the UNC nanoWorkbench [56] and the CSIRO Haptic Workbench [165], the VHW setup has the advantage of facilitating whole-arm haptic interaction and ambidextrous manipulation, using a wide-angle head-tracked visual display, and providing direct correspondence between the visual and haptic workspaces. Furthermore, the workbench surface is tilted at a 20 degree angle, which increases the visual range and aligns the hotspots of the visual and haptic workspaces [122]. Placing the PHANToM arm in front of the projection screen has the disadvantage of occluding the view and further reducing the extent of the available stereoscopic workspace. A related problem is that the low stiffness of the arm is apparent during hard surface contact simulation, because the PHANToM end-effector visually penetrates the graphical representation of the surface. To alleviate these problems, a fixed offset is added between the actual endpoint and its graphical representation. This is usually not a problem,

because users can tolerate a small offset between the physical and virtual interaction points for typical navigation and manipulation tasks [135]. Mounting the haptic device behind the screen would completely solve the occlusion problems, but such configuration is possible with front-projection screens only. Using a front-projection display, however, would further reduce the size of the visual workspace, because the projection surface has to be located closer to the eyes of the user.

3.2 User Interaction Techniques

One of the grand challenges of using immersive environments for scientific data exploration is making interaction comfortable, fast, and effective [171]. Designing and evaluating interaction techniques and user interfaces is an important area of virtual environment research [14]. Even though immersive virtual reality provides the possibility for more natural interaction, working with a computer generated 3D world is difficult, because the haptic cues that are part of the real world are missing from these environments. In the past, a variety of complex interaction techniques have been developed for immersive virtual reality applications. Some of these are not particularly suitable for everyday use. In contrast, the desktop WIMP (Windows, Icons, Menus, Pointers) paradigm has been very successful due to its simplicity, robustness, and convenience.

Figure 3.2 shows an example of a scientific visualization application running on the Visual Haptic Workbench. Scientific visualization applications use direct manipulation widgets for controlling visualization parameters, as shown in Figure 3.2. For the VHW setup, Pinch Gloves are used primarily for picking and manipulating virtual objects in the scene. Direct picking is not possible for objects located behind the screen. Ray-casting or aperture-based methods [47] can be used instead, of which the latter yields more intuitive and predictable manipulation. It is also possible to use the gloves for controlling cascadig two-dimensional menus [16]. However, most people who tried the system preferred using pointing instead of pinching gestures to switch between menu levels. Constrained interaction on the surface of the workbench is one possibility for increasing precision and reducing fatigue [109]. However, the success of this approach is largely incluenced by the registration accuracy of the system components. As shown in Chapter 4, it is possible to reduce geometric registration errors to a few millimeters for the VHW system.

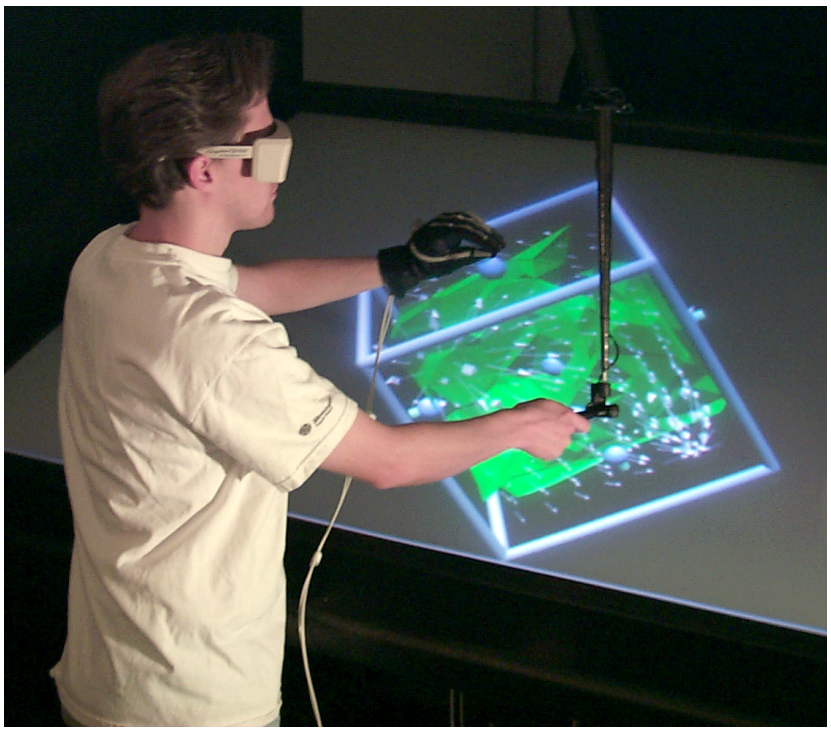


Figure 3.2. Example of an immersive scientific visualization application running on the Visual Haptic Workbench. The user is provided with a 3D stereoscopic view of the data and can interactively manipulate the visualization interface and probe the data using passive and active input devices.

Overall, the following guidelines were found useful for developing interaction techniques for immersive visualization applications running on the Visual Haptic Workbench [26, 162]:

- Avoid complex and encumbering devices, such as gloves.
- Use intuitive and simple interaction metaphors; reserve "magic" techniques for expert use, including shortcuts or text input [58, 15].
- Utilize two-handed manipulation when applicable, but provide ways to perform the same task with a single hand.
- Use physical and virtual constraints to increase precision and reduce fatigue.

Constraint-based haptic feedback, presented in Chapter 6, provides a promising approach for improving the precision of manipulation of user interface elements on the Visual Haptic Workbench.

3.3 Software Framework

Creating successful immersive applications is inherently an iterative process. An active area of virtual environment research is making application development comfortable, fast, and effective. Application development and evaluation, however, usually happen in two different workspaces. Ideally, the developer should be able to run an application on every available platform without modifications, using an interface optimized to that particular platform [87]. For example, Figures 3.3 and 3.4 show the same visualization application running on the VHW and in a desktop environment. Previous efforts on creating an immersive tool that could also run on the desktop required significant contributions from the programmer, because the interface remained platform dependent [141]. Currently, most frameworks do not support the concept of application transparency and provide a simulator mode instead [97, 11, 34]. In this mode, a third-person view of the user is presented such that immersive controls are mapped to a 2D desktop interface. Even though this mode is useful for examining how the user's actions affect the environment and vice versa, it prevents the developer from focusing on the content of the application.

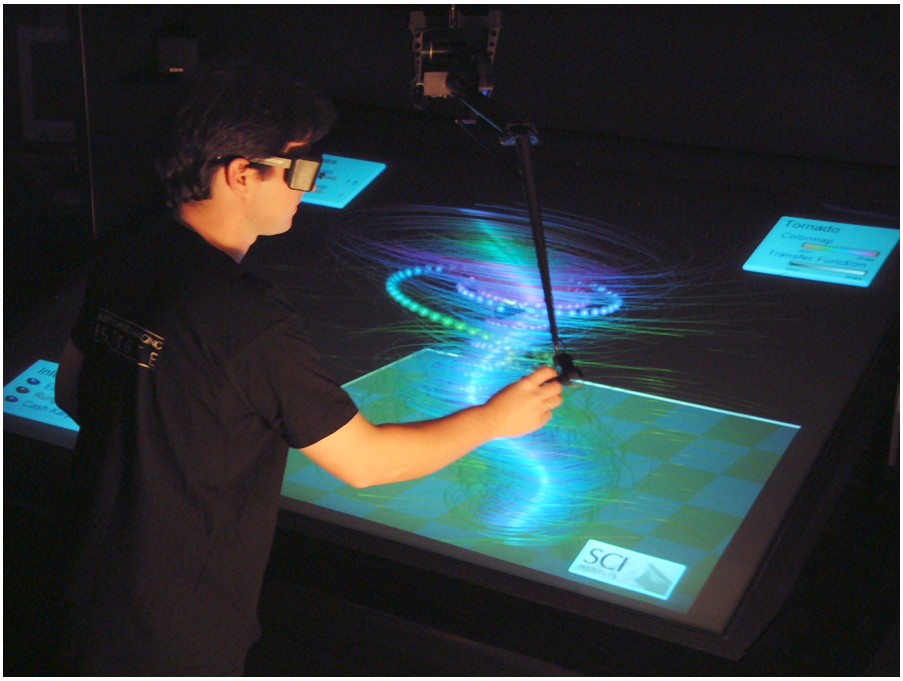


Figure 3.3. A user explores a tornado data set on the Visual Haptic Workbench.

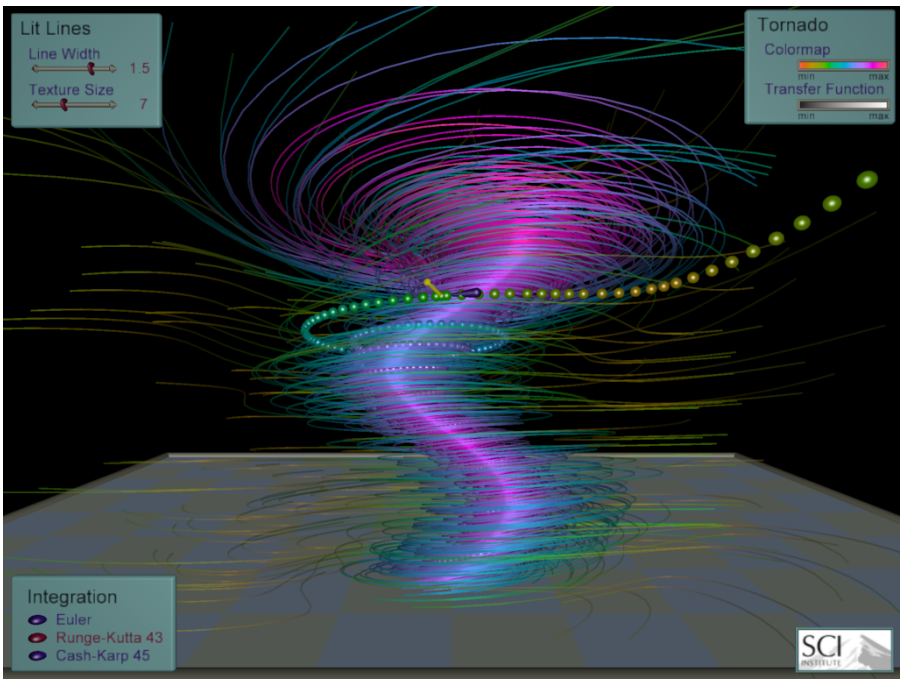


Figure 3.4. Screenshot of the tornado application running in a desktop environment.

The software development framework for the Visual Haptic Workbench was designed with the above considerations in mind. The core of the framework is a multithreaded rendering kernel and a simple but flexible configuration file format and parser. The structure of the core is similar to VRJuggler [11], but the implementation is considerably simpler. The kernel takes care of device initialization, creates the rendering window, and runs the simulation loop, which interfaces with the application through callbacks. Haptic rendering is supported via an extended version of the BasicIO library [159], which allows the servo loop to run at arbitrary frame rates and perform custom kinematic and force calculations. Applications can be created from a template class and are compiled into shared libraries. This way they can be dynamically loaded and exchanged at run-time without having to restart the kernel. Communication between device and application threads is provided via a common shared memory area. To facilitate transparent and clean device data representation, four basic atomic types are available: a discrete type for buttons, a continuous type for dials and mouse motion, a pose type for position trackers, and a force type for haptic devices. Atoms are aligned in memory at cache line boundaries and have an associated high-resolution time stamp. Multiple buffers are used for inter-

thread communication to minimize operating system overhead [138]. The interpretation of the device data is delegated to the interaction technique, as opposed to creating a unified device layer [62]. Desktop and immersive interaction techniques have separate implementations but provide a common programming interface. To reduce rendering load, intersection testing is implemented in software instead of relying on the OpenGL picking and selection mechanism [185]. In addition, graphical components including geometry and text elements are available for creating 3D user interfaces. A scenegraph could be used for structuring the components [166]. However, it is often not needed for simple visualization applications. To provide comfortable interaction, widget behavior have to be optimized for a particular platform. A suite of custom visualization techniques and a class hierarchy for scalar, vector, and diffusion tensor data on uniform and tetrahedral grids are also part of the framework. Traditionally, the visualizaion techniques are connected in a data-flow pipeline [155]. However, real-time immersive presentation and manipulation of data presents challenges for the dataflow paradigm, because each component is either static or has to act rapidly in response to input from the user.

CHAPTER 4

CALIBRATION AND REGISTRATION TECHNIQUES FOR THE VISUAL HAPTIC WORKBENCH

In this chapter several techniques are presented that can be used for improving the registration accuracy of combined immersive visual and haptic displays. In particular, the results demonstrate that it is possible to reduce the static geometric error present in the Visual Haptic Workbench by calibrating and colocating the system components. Ideally, the goal is to reduce the error of the position trackers and display parameters, so an overall registration accuracy in the order of a few millimeters is obtained. For the VHW setup, even in the highest resolution display mode the size of a single pixel is over one millimeter. Thus, errors in the order of a few millimeters result in at most one or two pixel errors in the displayed image.

The organization of the chapter is the following. First, a calibration framework for position tracking devices is developed and evaluated using the workbench hardware. Next, a technique is presented for calibrating the PHANToM using a simple measurement apparatus. Following this approach, a method is developed for colocating the tracking components of the system. Finally, an analysis is provided for quantifying the visual distortion caused by viewing errors present in head-tracked projection display systems.

4.1 Geometric Model of the Visual Haptic Workbench

Figure 4.1 illustrates the location of the components of the Visual Haptic Workbench setup. Immersive applications rely on several frames of reference in the system, including the tracking system $\mathbf{R}_0$, the screen location $\mathbf{S}$, the PHANToM base $\mathbf{P}_0$ and end-effector $\mathbf{P}_6$, the head and eye locations $\mathbf{H}$, $\mathbf{e}_L$ and $\mathbf{e}_R$, the stylus tip $\mathbf{U}$ located relative to the sensor frame $\mathbf{R}_1$, and the pinch spot $\mathbf{V}$ located relative to the sensor frame $\mathbf{R}_3$. Transformations

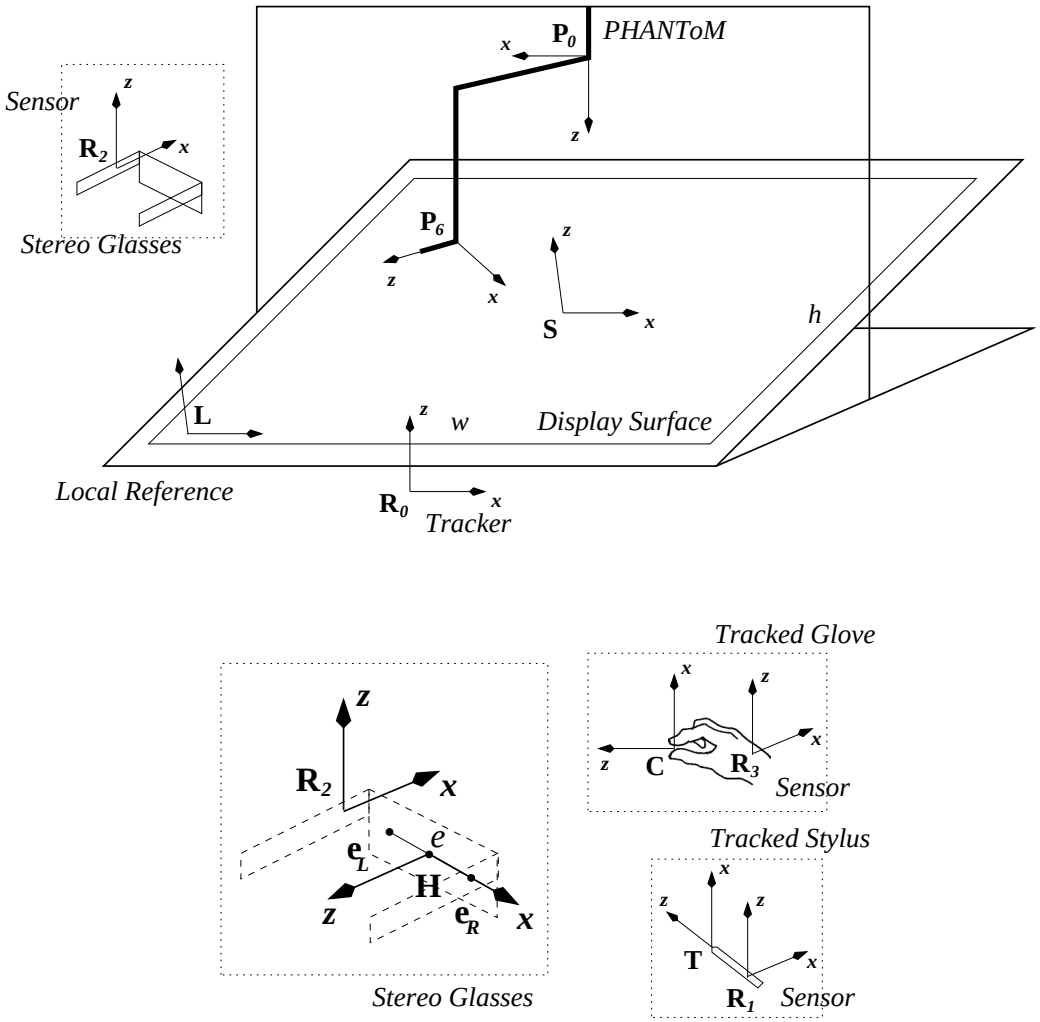


Figure 4.1. Geometric model of the Visual Haptic Workbench. Possible sources of error include individual device workspace distortions and misalignments of the respective coordinate systems.

$\mathbf{f}_{T_i} = \mathbf{T}_{R_0 \to R_i}$ and $\mathbf{f}_P = \mathbf{T}_{P_0 \to P_6}$ represent the position and orientation measurements for the tracker sensor $\mathbf{R}_i$ and the PHANToM endpoint $\mathbf{P}_6$, respectively.

The possible geometric error sources for the workbench setup include tracker distortions and the unknown rigid-body transformations between the fixed coordinate frames in the model. Device inaccuracies are captured by measurement errors $\Delta\mathbf{f}_{T_i}$ and $\Delta\mathbf{f}_P$, respectively. Calibration techniques aim to reduce measurement discrepancies by characterizing the error of the underlying mechanism of pose measurement. For magnetic tracking devices calibration involves finding a suitable parametric description of the magnetic field distortion and a way of extracting the model parameters from a number of

calibration measurements [76]. Alternatively, a lookup table (LUT) can be constructed that captures most of the error [110]. For haptic devices, the model already exists, but it is typically augmented with additional parameters, such as joint gains and offsets [67].

Coregistration is concerned with finding the remaining unknown display parameters, including the relative transformations between the fixed coordinate systems. Some of these parameters can be measured accurately, for example, the screen width w and height h. Others might need to be controlled explicitly, such as the eye separation distance e for bringing objects into the fusible stereoscopic range [180]. Some parameters are generally not very accurately known, such as the head reference frame $\mathbf{H}$, the projection centers in the eyes $\mathbf{e_L}$ and $\mathbf{e_R}$, and the interaction device hotspots. For improved interaction it is desirable to obtain user-specific measurements of these parameters. Typically, coregistration methods assume that accurate tracking is available [53], which is not the case for the VHW configuration.

4.2 A Calibration Framework for Position Tracking Devices

A significant component of the overall registration error is attributed to the position tracking system. The following sections develop a framework for improving and characterizing the accuracy of position tracking devices. Even though the framework is evaluated using an electromagnetic tracking system, it provides a solution to the problem of finding and approximating the distortion field of any position tracking system in general.

4.2.1 The Calibration Problem

The goal of position tracker calibration is to measure and characterize the relationship between the actual tracking sensor position $\mathbf{p}_a$ and orientation $\boldsymbol{q}_a$ and those reported by the tracker, $\mathbf{p}_t$ and $\boldsymbol{q}_t$, as illustrated by Figure 4.2. Typically, calibration is performed by placing the sensor at a number of known locations in the tracked workspace and measuring its position and orientation, $\mathbf{p}_m$ and $\boldsymbol{q}_m$, with respect to a common reference frame attached to the tracking system. The measured pose either closely matches the actual pose or their difference is small enough relative to the distortion that they can be considered equal, $\mathbf{p}_m = \mathbf{p}_a$ and $\boldsymbol{q}_m = \boldsymbol{q}_a$.

Position error is generally defined as the vector difference between the measured and reported positions:

$$\mathbf{p}_e = \mathbf{p}_m - \mathbf{p}_t \tag{4.1}$$

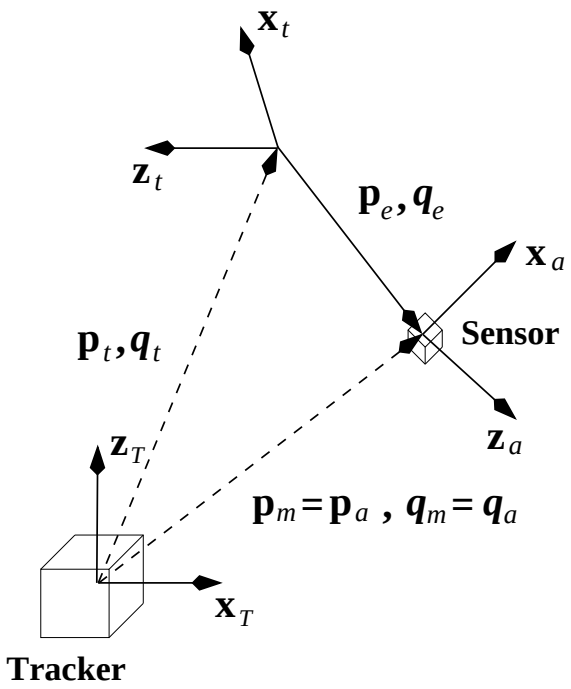


Figure 4.2. Definition of calibration error. Calibration error is defined as the difference between the measured and reported sensor position and orientation.

Orientation error is not as straightforward to define, because there is no representation that is both minimal and provides a nonsingular metric for optimization. Euler angles are minimal but suffer from singularities, while rotation matrices and quaternions are redundant and subject to parameter constraints. Orientation errors observed with magnetic tracking systems, however, are bounded by a fairly general limit of 30 degrees, which implies that a reduced representation can be used for calibration.

4.2.2 Representation of Orientation Errors

Orientation errors can be formulated from the rotation matrix or quaternion that expresses the difference between the reported and measured orientation of the sensor [92]. An alternative approach obtains Euler angle corrections by sampling the sensor in the reference orientation [91]. In this case, the rotation matrix computed from the correction angles is multiplied by the reported orientation matrix to yield the corrected orientation matrix.

In previous work, orientation error was defined as the quaternion difference between the closed loop transformation between the sensor and the tracking reference frame and the identity quaternion [110]. The outcome of these experiments suggested that for

magnetic tracking systems the orientation error depends on not only the position of the sensor but also its orientation. However, careful investigation reveals that the error definition is orientation dependent, since a given rotation in the reference frame has a different form in the sensor frame depending on the orientation of the sensor. For example, a rotation about the **x** axis in the reference frame remains the same rotation in the sensor frame when the two frames are oriented in the same way, but becomes a rotation about the **y** axis when the sensor is rotated by 90 degrees about the **z** axis of the reference.

To overcome the problem of orientation dependence, the orientation error is defined using quaternion composition and rotation [100]. The measured and reported orientations are related by orientation error quaternion ${}^{t}\boldsymbol{q}_e$:

$$\boldsymbol{q}_m = \boldsymbol{q}_t \, {}^{t}\boldsymbol{q}_e \tag{4.2}$$

where $\boldsymbol{q}_m$ and $\boldsymbol{q}_t$ are expressed in the tracking reference frame, and ${}^{t}\boldsymbol{q}_e$ is expressed in the sensor frame, as shown in Figure 4.2. Note that the error given by equation 4.2 is orientation dependent. To obtain an orientation independent error quaternion, the error $\boldsymbol{q}_e$ is expressed in the fixed reference frame:

$$\boldsymbol{q}_e = \boldsymbol{q}_t \, {}^{t}\boldsymbol{q}_e \, \boldsymbol{q}_t^* = \boldsymbol{q}_t \, (\, \boldsymbol{q}_t^* \, \boldsymbol{q}_m \,) \, \boldsymbol{q}_t^* = \boldsymbol{q}_m \, \boldsymbol{q}_t^* \tag{4.3}$$

where $*$ denotes the quaternion conjugate operator.

Assuming the calibration procedure yields error correction quaternion $\boldsymbol{q}_e'$, the corrected orientation can be computed by quaternion composition:

$$\boldsymbol{q}_t' = \boldsymbol{q}_t \, (\, \boldsymbol{q}_t^* \, \boldsymbol{q}_e' \, \boldsymbol{q}_t \,) = \boldsymbol{q}_e' \, \boldsymbol{q}_t \tag{4.4}$$

Equations 4.3 and 4.4 provide a simple technique for computing the orientation error from the experimental measurements and applying the reconstructed error for correction. In contrast, when using Euler angles, one has to convert them to a rotation matrix, extract the rotation axis, express it in the sensor frame, and then compute the rotation correction matrix. Even when exploiting the composition suggested by 4.4, one still has to convert the Euler angles to a rotation matrix.

A disadvantage of quaternions is that they are redundant. For example, quaternions have to be normalized when computed from interpolation or an analytical model to be applied correctly [92]. Note, however, that experimental measurements indicate that orientation errors are relatively small. Thus, the scalar part of $\boldsymbol{q}_e$ is always close to one.

Hence, the vector part of $\boldsymbol{q}_e$ provides the desired minimal representation and the scalar part can be computed from the constraint of unity [85].

The quaternion vector part is similar to other representations used in robotics and computer animation. Remember that $\boldsymbol{q}_e$ is related to the angle and axis of rotation by:

$$\boldsymbol{q}_e = q_e + \mathbf{q}_e = \cos(\frac{\theta}{2}) + \sin(\frac{\theta}{2})\,\mathbf{k} \tag{4.5}$$

If the rotation error angle θ is small, the vector part can be approximated by:

$$\mathbf{q}_e = \sin(\frac{\theta}{2})\,\mathbf{k} \approx \frac{\theta\,\mathbf{k}}{2} \tag{4.6}$$

The above approximation is a scaled version of the exponential map [57] and is related to differential orthogonal rotations [67].

4.2.3 Calibration via Polynomial Fit

Polynomial fit techniques were among the first calibration methods applied and evaluated for correcting magnetic tracker distortion [25]. Among their advantages are simplicity, robustness, and speed, since only a few parameters need to be stored for reconstructing the error at a given location. Their disadvantage is that they provide a poor fit for small errors near the transmitter and are not as accurate at capturing local field distortions as lookup table based methods.

The polynomial coefficients are obtained from a least squares fit that minimizes the difference between the measured errors and those predicted by the calibration model. A degree r vector polynomial of position $\mathbf{p}$ can be formulated as:

$$\mathbf{f}(\mathbf{c},\mathbf{p}) = \mathbf{f}(\mathbf{c},x,y,z) = \sum_{j=1}^{R} \begin{bmatrix} c_{x,j} \\ c_{y,j} \\ c_{z,j} \end{bmatrix} x^{s_j}\, y^{t_j}\, z^{u_j} \tag{4.7}$$

The formulation assumes that each term in the polynomial is unique, at most degree r with nonnegative powers ($0 \le s_j + t_j + u_j \le r$), and the polynomial is complete, meaning that all combinations of powers are present in the sum. Given these conditions, it can be shown that the number of polynomial terms in equation 4.7 is:

$$R = \frac{(r+1)(r+2)(r+3)}{6} \tag{4.8}$$

For example, a degree four polynomial has 35 unique terms.

The goal of the calibration procedure is to find a coefficient vector $\mathbf{c}$ such that the resulting polynomial closely approximates the error at the measurement locations. For position errors this can be written as the minimization of the following objective function:

$$S_p = \sum_{i=1}^{N} \|\mathbf{p}_e^i - \mathbf{f}(\mathbf{c}_p, \mathbf{p}_t^i)\|^2 \tag{4.9}$$

where N is the number of collected data points, $\mathbf{p}_t^i$ is the position reported by the tracker, and $\mathbf{p}_e^i$ is the position error at measurement location i defined by equation 4.1. This is a linear least squares problem and has a well known closed-form solution [139]. Note that equation 4.9 is an over-determined system of equations if and only if $N > R$, meaning that more measurements are needed than the number of polynomial terms for minimizing S_p.

For orientation error, the parameter estimation problem is formulated as finding the minimum of:

$$S_q = \sum_{i=1}^{N} \|\Delta\mathbf{q}^i\|^2 = \sum_{i=1}^{N} \|\mathbf{q}_e^i - \mathbf{f}(\mathbf{c}_q, \mathbf{p}_t^i)\|^2 \tag{4.10}$$

where $\mathbf{q}_e^i$ is the vector part of the error quaternion obtained from Equation 4.3 at measurement location i. Hence, orientation errors are treated in a similar way to position errors. Alternative definitions for $\Delta\mathbf{q}^i$ include the vector part of the quaternion that rotates between the computed and measured error quaternions and the error vector obtained from differential orthogonal rotations. There is no significant difference between the results obtained from these alternative error definitions. The definition of Equation 4.10, however, is simpler, because it yields a linear problem, while the other methods require the use of iterative nonlinear parameter estimation algorithms [74].

4.2.4 Experimental Apparatus

A good experimental apparatus provides position and orientation measurements that are sufficiently accurate, so they do not degrade the efficacy of the error correction technique. The apparatus should cover the desired tracking space and not induce further distortions in the magnetic field. Optical measurement systems are the most suitable for this purpose; they do not use mechanical linkages, such as digitizing arms, and have higher accuracy than alternative technologies, such as ultrasonic range measurement. In addition, they can be used to easily establish a precise transformation between the tracker frame and the frame of the display surface.

An NDI Optotrak 3020 position tracking system was used for the experimental evaluation of the technique. The Optotrak measures the position of active infrared markers with a nominal accuracy of 0.05 mm per meter of distance along the central focal axis [129]. A nonmetallic platform was constructed for mounting the magnetic sensor along with six Optotrak markers, as shown in Figure 4.3. The markers were mounted in positive and negative directions along approximately orthogonal axes, such that at least three markers were visible by the camera system at a time. The platform was designed to be mobile with rotational redundancy, such that it was possible to reach arbitrary orientations throughout the workspace. During the experiment, samples were taken independently from the magnetic tracker and the Optotrak. It was also verified that the placement of the markers did not distort the magnetic field. Two numerical procedures were developed in support of the apparatus: one for estimating the fixed geometric relationship between the transmitter and the Optotrak, and the location of the markers relative to the sensor, and another for extracting the sensor pose from three simultaneous optical position measurements [74].

The measurement apparatus can be described by a kinematic model shown in Figure 4.4. The model relates the actual sensor pose to the marker positions, which is expressed by the following set of equations:

$$\mathbf{v}_j = \mathbf{p}_0 + \mathbf{R}(\boldsymbol{q}_0)(\mathbf{p} + \mathbf{R}(\boldsymbol{q})\,\mathbf{r}_j) \quad j = 1 \ldots 6 \tag{4.11}$$

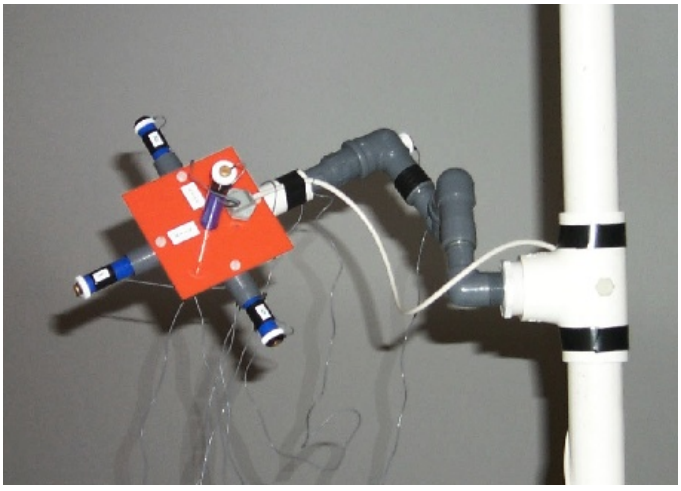


Figure 4.3. Experimental platform used for data collection. Markers are mounted in six directions around the magnetic tracking sensor.

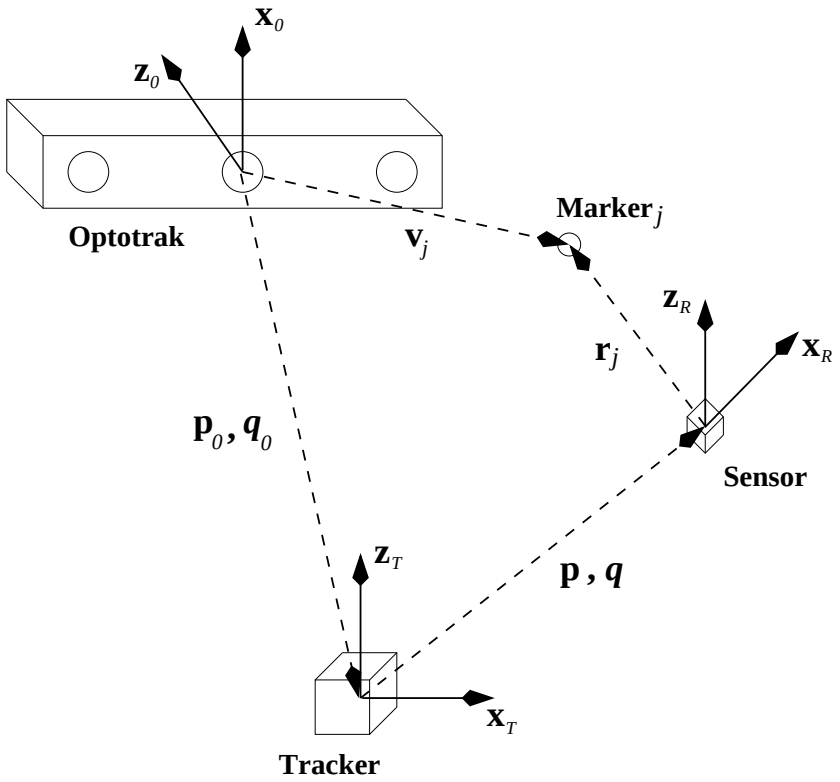


Figure 4.4. Geometric model of the calibration apparatus.

where $\mathbf{p}_0$ and $\boldsymbol{q}_0$ define the transformation between the Optotrak and tracking reference frames, $\mathbf{r}_j$ is the location of marker j in the sensor coordinate system, $\mathbf{p}$ and $\boldsymbol{q}$ are the actual position and orientation of the sensor, $\mathbf{v}_j$ is the position of marker j as a function of the sensor pose, and $\mathbf{R}(\boldsymbol{q})$ is an operator which converts a quaternion to a rotation matrix [101]. Only three marker measurements are used at a particular location, yielding a reduced set of equations:

$$\mathbf{y}^i = \begin{bmatrix} \mathbf{v}^i_{j_1} \\ \mathbf{v}^i_{j_2} \\ \mathbf{v}^i_{j_3} \end{bmatrix} = \mathbf{y}^i(\boldsymbol{\phi}, \mathbf{p}^i, \boldsymbol{q}^i) \quad i = 1 \ldots N \tag{4.12}$$

where $j_1, j_2, j_3 \in [\,1 \ldots 6\,]$ are the indices of the markers that are visible by the Optotrak. The fixed geometric parameters are collected into parameter vector $\boldsymbol{\phi}$:

$$\boldsymbol{\phi} = \begin{bmatrix} \mathbf{p}_0 \\ \boldsymbol{q}_0 \\ \mathbf{r}_1 \\ \vdots \\ \mathbf{r}_6 \end{bmatrix} \tag{4.13}$$

Since $\boldsymbol{\phi}$ is not known in advance, it is estimated by registering the transmitter location to the Optotrak and the marker positions in the sensor frame. Thus, 20 extra measurements were collected in close proximity of the transmitter where the field distortions could be assumed to have a minimal effect on the parameter estimates. Since the optical measurements are more accurate than the data collected from the tracker, a version of a total least squares estimation technique [157] was implemented, which takes into account the input noise present in the system described by equation 4.12. The validity of the registration results were verified via the χ^2 statistic [139] obtained after the fit, which indicated that the *a priori* assumptions about the noise model were appropriate. The *a posteriori* covariance matrix of the parameter estimates were also examined. The estimation error had a standard deviation of 0.15 mm, which is the nominal accuracy of the Optotrak measurements for the VHW setup.

In general, using quaternions in both parameter vector $\boldsymbol{\phi}$ and input measurements $\boldsymbol{q}^i$ causes problems with the estimation procedure, because they have to be renormalized at every iteration. Unfortunately, the renormalization can prohibit convergence of the algorithm. The problem can be avoided by including the calculation of the largest magnitude component in the kinematic equations. This is accomplished by operator $\boldsymbol{q}(\mathbf{s})$ that converts three quaternion elements to a unit quaternion:

$$\mathbf{v}_j = \mathbf{p}_0 + \mathbf{R}(\boldsymbol{q}(\mathbf{s}_0))(\mathbf{p} + \mathbf{R}(\boldsymbol{q}(\mathbf{s}))\,\mathbf{r}_j) \quad j = 1 \ldots 6 \tag{4.14}$$

It can be shown that the Jacobian of the operator $\partial\boldsymbol{q}/\partial\mathbf{s}$ is well-conditioned as long as $\mathbf{s}$ is the minimum magnitude part of $\boldsymbol{q}$ [76]. To satisfy this condition, the system is dynamically reparameterized at every iteration of the estimation procedure, which is similar to the method suggested for computer animation using the exponential map [57].

After the vector of fixed geometric parameters $\boldsymbol{\phi}$ is found, the position $\mathbf{p}^i$ and orientation $\mathbf{s}^i$ of the sensor at measurement location i is computed from the three marker positions $\mathbf{y}^i$ by computing the inverse of equation 4.12. Although an analytical solution exists, it is unclear how it behaves numerically when the location of the markers are not known precisely. Instead, an iterative procedure is used that finds the pose by minimizing the error between the computed and measured marker positions. Good initial estimates are provided by the original tracker readings and convergence is achieved in a few iterations.

4.2.5 Experimental Results

A total of 600 measurements were collected within a $1.8\,\mathrm{m} \times 1.5\,\mathrm{m} \times 0.9\,\mathrm{m}$ volume located above and in front of the display surface of the workbench. Possible sources of distortion included metal reinforcement in the floor 0.8 m below the transmitter, a metal door 2.1 m away from the transmitter, and several CRT displays, each located at least 1.8 m away from the transmitter. The collection of 480 calibration poses was based on a $12 \times 10 \times 5$ grid with a 0.15 m cell size. In addition, 120 validation measurements were taken randomly in the workspace. The orientation of the sensor was randomized to ensure that it covered the space of possible rotations. Only two calibration poses had to be rejected because of unreasonably large deviations in the optical measurements.

Degree four polynomials were fit to the position and orientation errors computed from the calibration measurements using the method described in Section 4.2.3. The success of the polynomial fit was evaluated using the validation data set. Statistical measures were computed from the position error magnitudes and orientation correction angles before and after calibration. These measures are collected in Table 4.1. The results indicate that position errors have been reduced by almost 90 % on average, whereas for orientation errors the reduction is slightly less, around 80 % on average.

Figure 4.5 shows scatter plots of the error magnitude as a function of transmitter-sensor separation distance. The figures indicate that overall the polynomial fit correction is quite effective, but does not improve accuracy within a 0.5 m radius volume of space around the transmitter. This problem is attributed to the global nature of the correction

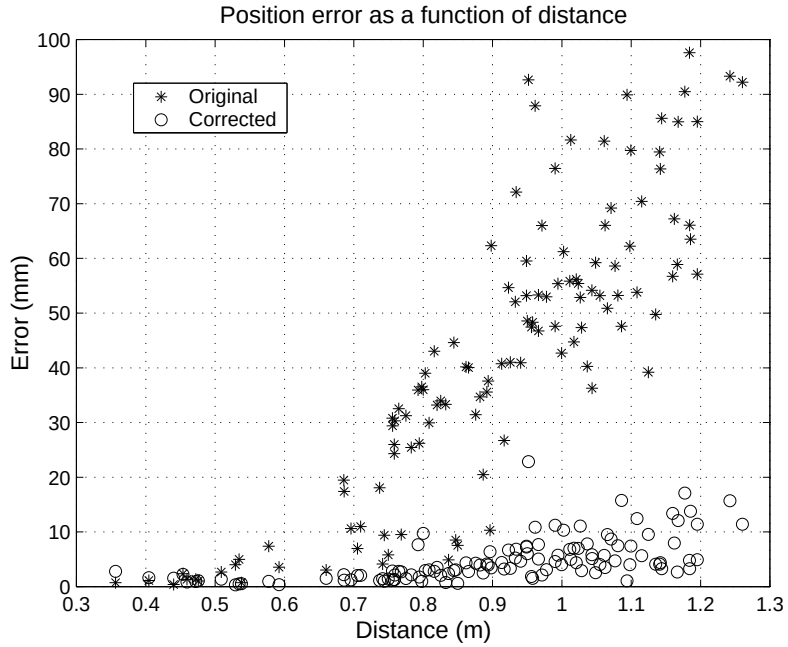
Table 4.1. Results of the calibration experiment.

Position error

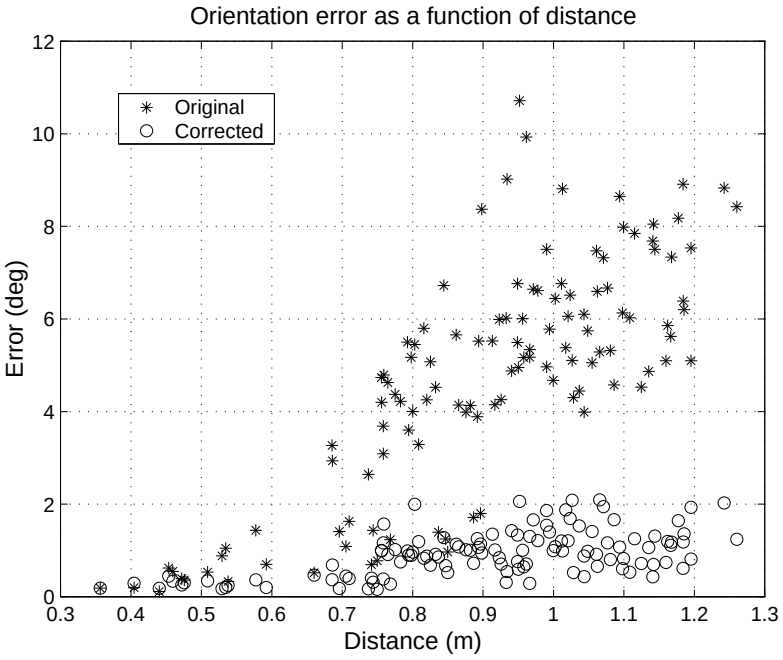
Measure (mm)	Original	Corrected	Improvement
Mean	42.3	4.82	88.6 %
Standard dev.	26.3	4.04	84.7 %
Maximum	97.5	22.8	76.6 %

Orientation error

Measure (deg)	Original	Corrected	Improvement
Mean	4.72	0.93	80.3 %
Standard dev.	2.51	0.50	80.1 %
Maximum	10.71	2.09	80.5 %



(a)



(b)

Figure 4.5. Tracking error as a function of distance from the transmitter. (a) Position error with and without correction. (b) Orientation error with and without correction.

technique [25, 91]. However, the results suggest that within the typical workspace of the sensors, it is possible to reduce tracking errors within a centimeter.

4.2.6 Visualization of Field Distortion

Several methods have been used in the past for visualizing the effect of magnetic field distortion on the position error, including iconic representations [25, 110] and grid visualization [187, 43]. Scatter plots and histograms provide quantitative information about the distortion field, but tell nothing about its shape and structure. Since all of the proposed techniques are based directly on the collected data, the quality of the visualization depends on the resolution of the sampled space.

Illuminated streamlines, used as a global visualization technique with additional local shading information, clearly show the structure of the position distortion field [191]. After selecting a number of seed points randomly in the x-y plane of the transmitter, streamlines were advected in the direction of the distortion vector field described by the error correction polynomial. The intensity and opacity of the line segments are varied according to the local magnitude of the field. Figure 4.6 shows the axially symmetric shape of the position distortion without visual clutter [25].

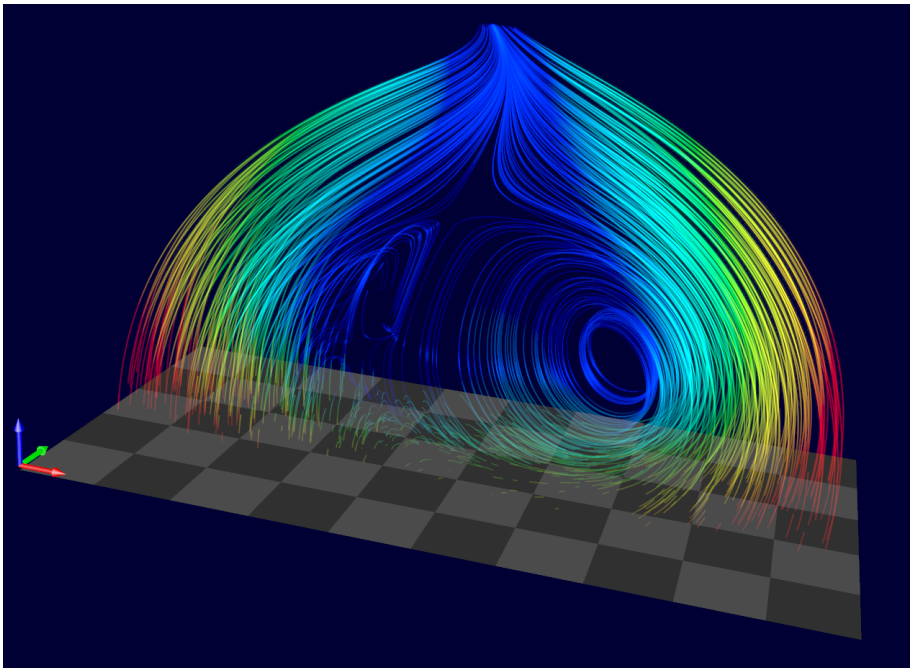


Figure 4.6. Visualization of position error. Illuminated streamlines are used to visualize the shape of the position distortion field.

Icon visualization was used to examine the structure of orientation distortion. In Figure 4.7, icons represent orientation error vectors at measurement locations in an x-y slice of the calibration data set. The magnitude of the icon is related to the orientation error angle via a logarithmic transfer function, and the direction of the icon corresponds to the axis of rotation. Icons with larger magnitudes form a circular pattern around the transmitter, indicating a functional relationship between the orientation error and the position of the sensor, independent of the orientation of the sensor. Smaller icons appear randomly oriented, probably because the error magnitude approaches the accuracy of the measurement apparatus near the transmitter.

Additional insight is obtained by visualizing the distortion of the generated magnetic field [100]. Visualization of the distortion field reveals important information about its structure and indicates potential sources of the distortion. As shown in Figure 4.8, the distortion field for the Ascension Flock of Birds magnetic tracker is very similar to a secondary magnetic field, possibly induced by the interaction of the primary field and a large metallic structure in the ceiling.

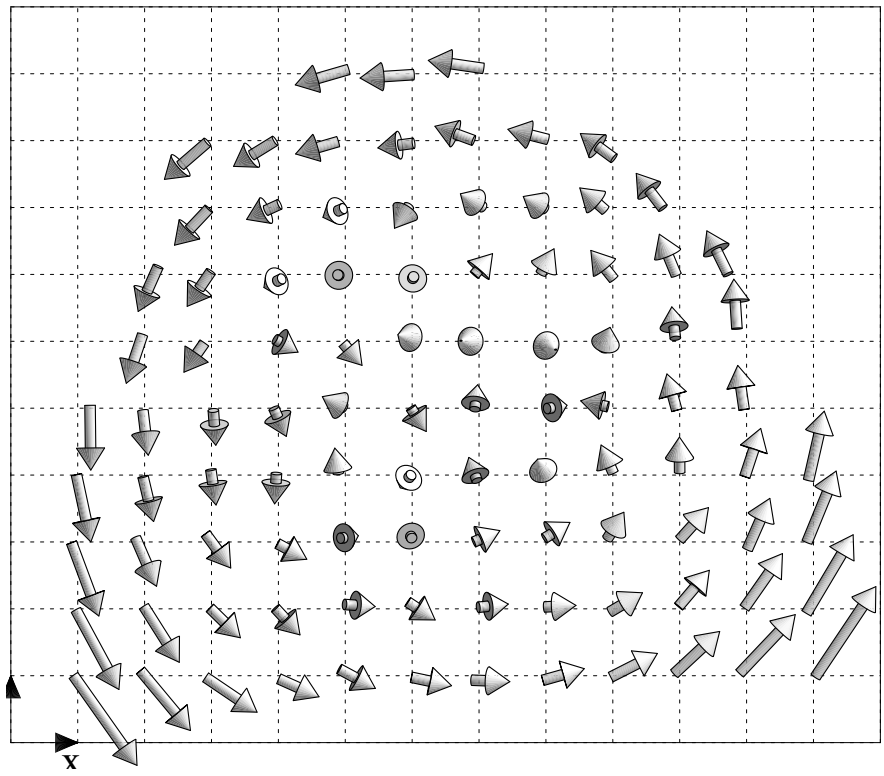


Figure 4.7. Visualization of orientation error. Icon visualization is used to show an x-y slice of the orientation error vectors collected near the transmitter.

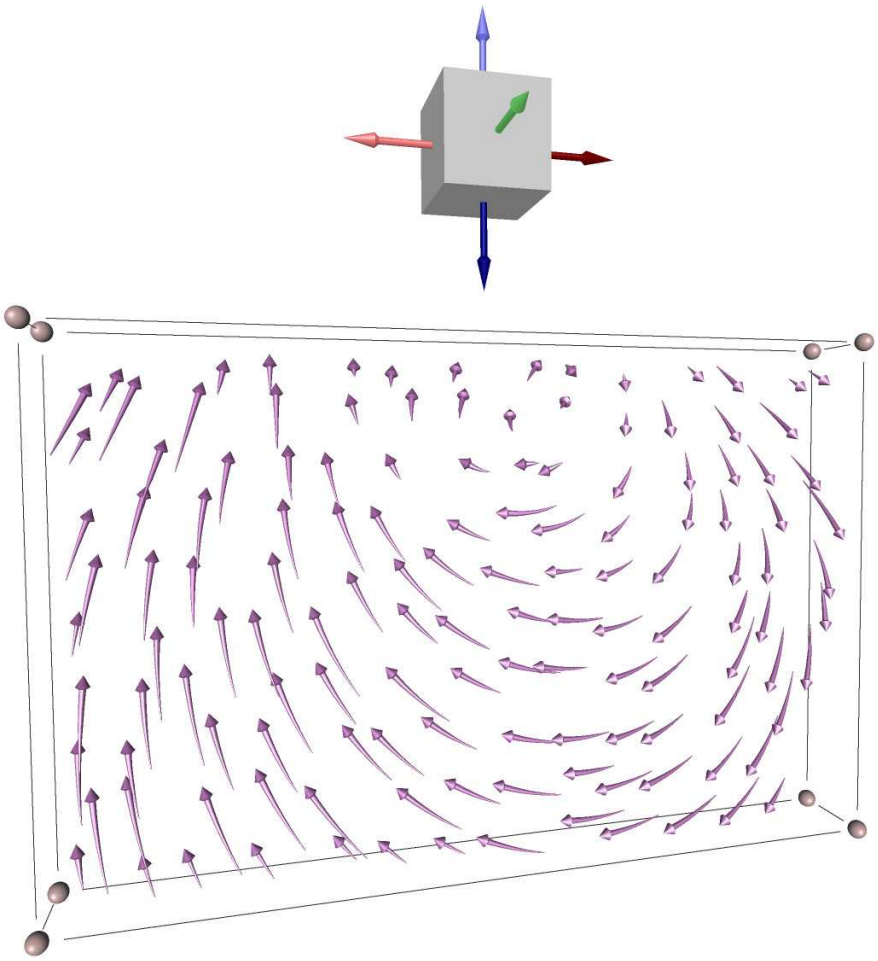


Figure 4.8. Visualization of magnetic field distortion. Icons visualization is used to show one slice of the distortion field around an Ascension Flock of Birds transmitter. The icons show that the distortion field has a magnetic moment in the **x** direction. The transmitter is indicated by the box on the top.

4.3 Calibration of the PHANToM

There are two reasons for the poor positioning accuracy of the PHANToM. First, the standard procedure for initializing the optical encoders is based on manually holding the arm in a reset position, as shown in Figure 4.9. This cannot be accomplished and repeated in a precise manner. Although a reset arm is available for initialization, it cannot be used in the overhead configuration, which is the configuration used for the Visual Haptic Workbench. Thus, a custom apparatus was constructed that allows the user to quickly place the end-effector to a fixed known location within the workspace of the device. Second, the pose and force calculations within the supporting libraries are

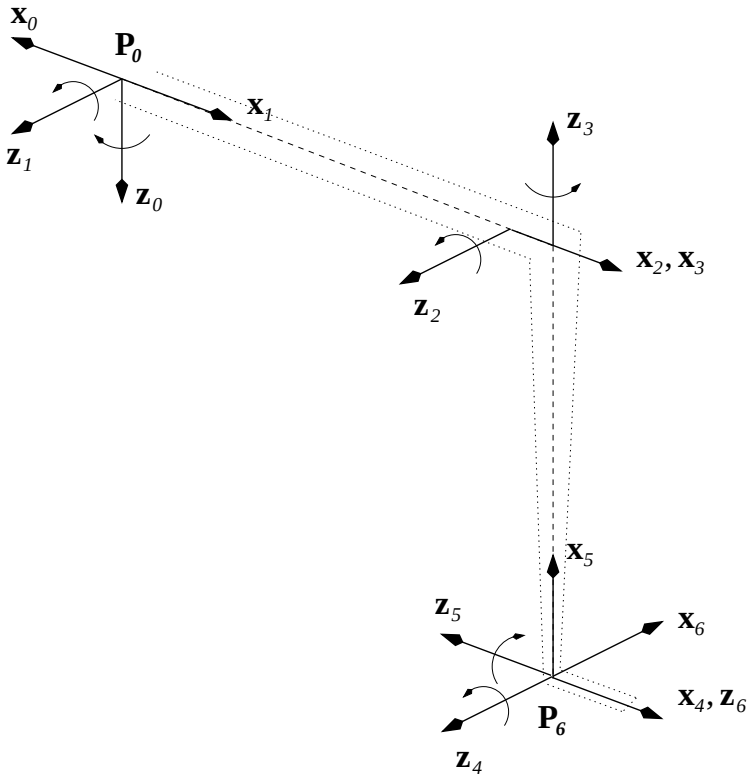


Figure 4.9. Kinematic model of the PHANToM 3.0L. The device is mounted in the overhead configuration. The stylus is augmented with a calibration tool for taking measurements on the display surface of the workbench.

based on the nominal kinematic model of the haptic interface, which is different from the actual realization. For example, the stylus end-effector of the device has visually apparent misalignment, because rotation axes $\mathbf{z_4}$ and $\mathbf{z_5}$ do not meet at a right angle. These error sources typically result in a total discrepancy of about $10 - 100$ mm and a few degrees at the stylus endpoint.

4.3.1 Kinematic Model

To find more accurate kinematic parameters, the device is calibrated by taking a number of joint angle measurements while positioning the endpoint on a grid placed on top of the display surface of the workbench. The PHANToM end-effector is augmented with a custom probe during calibration. The mixed Denavit-Hartenberg-Hayati parameters [67] for the complete system, including a base reference frame $\mathbf{P}_{-1}$ on the grid, are collected in Table 4.2. Values not estimated by the calibration procedure are indicated by a boldface

Table 4.2. Nominal parameters of the PHANToM setup. The kinematic model is described using Denavit-Hartenberg-Hayati notation. Distances are given in millimeters and angles are given in degrees. Fixed parameters are indicated in boldface.

j	d_j	a_j	α_j	β_j	γ_j
0	d_0	a_0	α_0	**0**	γ_0
1	d_1	0.0	-90.0	**0**	γ_1
2	**0**	457.2	0.0	0.0	0.0
3	0.0	0.0	-90.0	**0**	0.0
4	-457.2	0.0	90.0	**0**	0.0
5	0.0	0.0	-90.0	**0**	90.0
6	**0**	**0**	**180**	**0**	**90**
7	d_7	a_7	**0**	**0**	γ_7

font. The remaining parameters are grouped into parameter vector $\boldsymbol{\phi}$ and input to the calibration procedure together with joint angle measurements $\boldsymbol{\psi}$.

4.3.2 Calibration Procedure

The calibrated parameters are found via nonlinear least-squares estimation by iteratively minimizing the error between the Cartesian coordinates of the grid points and the calibration tool endpoint:

$$\Delta\mathbf{p}^i = \mathbf{p}_0^i - \mathbf{p}^i(\boldsymbol{\phi}^k, \boldsymbol{\psi}^i) = \frac{\partial \mathbf{p}^i}{\partial \boldsymbol{\phi}}(\boldsymbol{\phi}^k, \boldsymbol{\psi}^i)\ \Delta\boldsymbol{\phi} = \mathbf{J}^i\ \Delta\boldsymbol{\phi} \tag{4.15}$$

The calibration procedure finds parameter correction vector $\Delta\boldsymbol{\phi}$ at iteration step k, where $\mathbf{p}^i(\boldsymbol{\phi}^k, \boldsymbol{\psi}^i)$ represents measurement location i expressed in the surface frame, computed from the forward kinematics of the device using the current parameter estimate $\boldsymbol{\phi}^k$ and raw joint sensor readings $\boldsymbol{\psi}^i$.

4.3.3 Experimental Results

Measurements were collected at $5''$ intervals on an 8×5 grid placed on top of the workbench surface. The experimental setup is shown in Figure 4.10. During data collection the orientation of the measuring probe was randomly selected to ensure that it covered the range of possible joint motions of the wrist. Only 37 of the 40 points were reachable by the device. Data acquisition could be performed conveniently at a rate

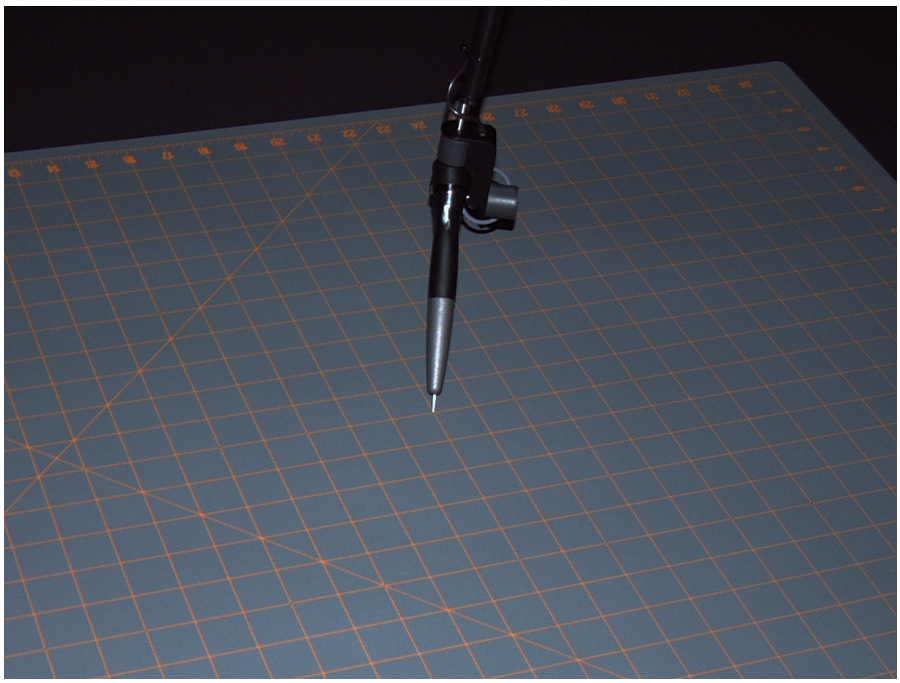


Figure 4.10. Experimental setup for calibrating the PHANToM. The PHANToM is calibrated by collecting joint angle measurements on a grid located on the surface of the workbench.

of three samples per minute. The estimated accuracy of the measurement apparatus is 0.5 ± 0.5 mm.

The location of the grid has a significant influence on the accuracy of the parameter estimation. The condition number of the aggregate parameter Jacobian matrix $\mathbf{J}$, assembled from the individual Jacobians from equation 4.15, is an indicator of the observability of the system [67]. The rule of thumb is that if the condition number is well over 100, the least-squares fit will not yield a reliable set of parameter estimates. This rule indicated that the location of the workbench screen was not appropriate for precise calibration, because the poor observability of the combination of several parameters yielded a condition number over 150. Hence, another grid was placed nearly perpendicular to the workbench surface, on which 20 more samples were collected. Note that the parameters given in Table 4.2 need to be augmented to account for the reference frame of the second grid. The combination of the two sets of measurements reduced the condition number to 70.

Figure 4.11 shows the deviations from the grid points using the nominal model, joint angle offset correction, and full calibration. Precise joint angle offsets significantly improve

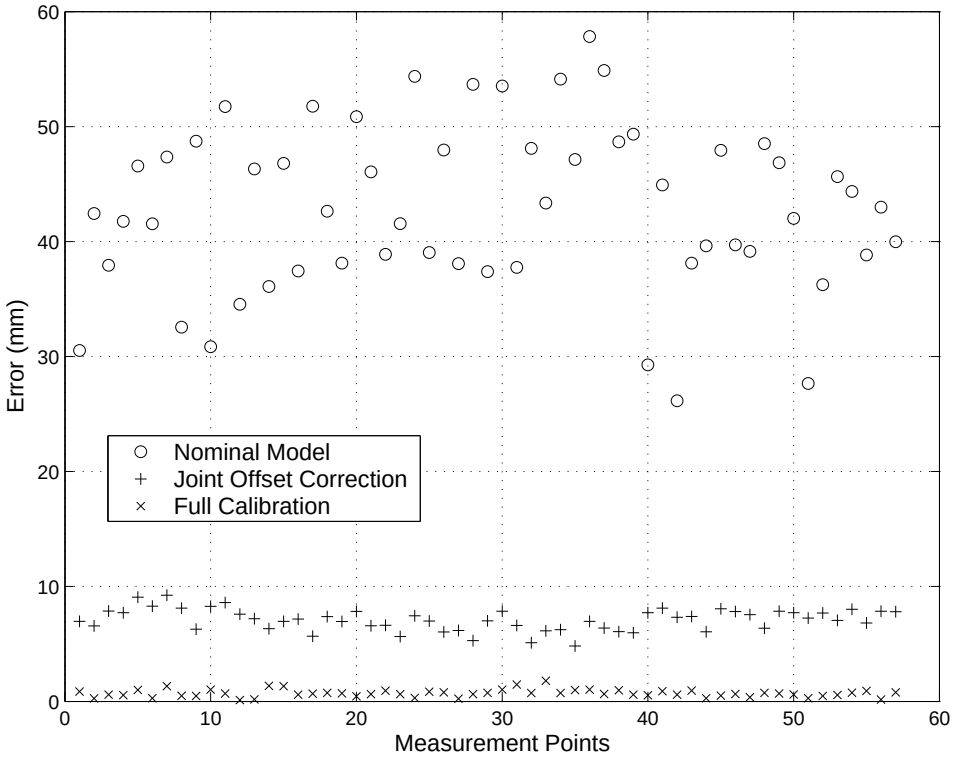


Figure 4.11. Position error between the tool endpoint and the calibration grid points. Notice the effect of correcting the joint angle offsets and how inaccurate the nominal model is.

the positioning accuracy of the device by reducing the error from $30 - 60$ mm to about 10 mm. Using the calibrated kinematic model reduces the error to about 1 mm, making the PHANToM an accurate position tracking device. A similar improvement was observed when omitting every third sample from the calibration data and using them for validating the approach.

4.4 Coregistration of Position Tracking Devices

To establish the transformation between the base coordinate frames of the tracking devices in the system, a registration technique similar to the one developed in Section 4.2.4 for colocating the magnetic tracker with the Optotrak was used. In this case, the goal was to find the coordinate transformation between the PHANToM base $\mathbf{P}_0$ and the tracker reference $\mathbf{R}_0$. By attaching a magnetic tracker sensor to the end-effector of the PHANToM using a rigid mechanical link, and taking a series of simultaneous measurements by

placing the link into a number of poses, it is possible to find the unknown rigid-body transformations between the coupled tracking devices [74].

There is a problem one has to be careful about when using the coregistration approach above. Since the last joint of the PHANToM is closely aligned with the axis of the link, full pose coregistration is not possible in this configuration. It is only possible to find the full transformation between the base coordinate systems and the displacement vector between the tracked endpoints, which reduces the total number of unknown parameters from 12 to 9. Also, since the last joint of the PHANToM has very little effect on determining the end-effector displacement, its motion has to be disabled during the data collection procedure.

To verify the feasibility of the approach, 25 measurements were collected with the apparatus shown in Figure 4.12. To obtain accurate readings from the magnetic tracker, the sensor was located in the vicinity of the transmitter during the experiment. Using this approach, the overall registration error between the devices was reduced to 1.5 mm, which corresponds to their nominal accuracy after calibration.

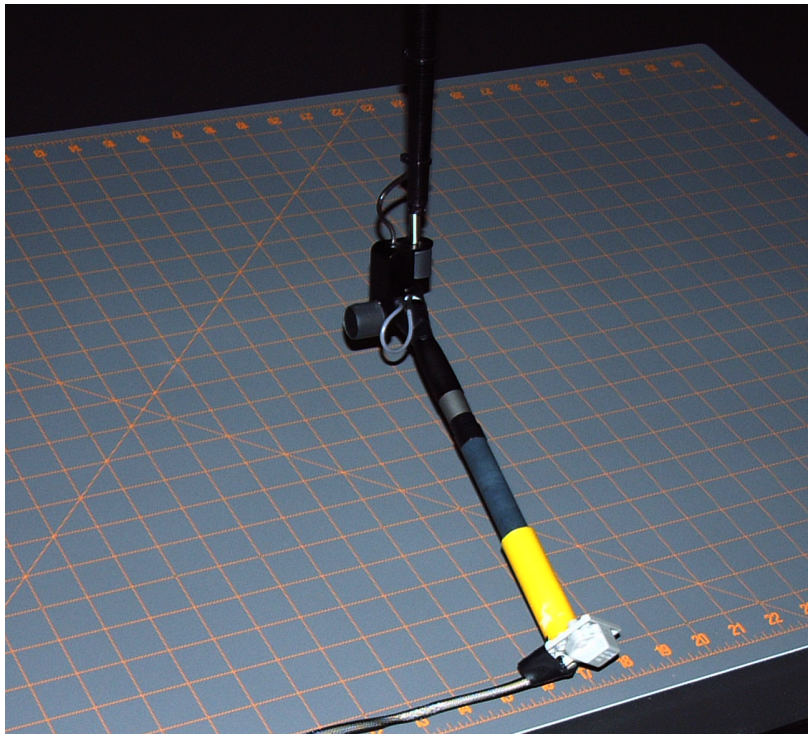


Figure 4.12. Experimental setup for registering the PHANToM to the position tracker. A tracker sensor is attached to the PHANToM end-effector using a rigid mechanical link.

4.5 Characterizing the Visual Distortion Caused by Head-Tracking Errors

A relevant issue that has only been partially investigated in the past is how viewing and tracking errors affect the perceived virtual world. It is well known that head orientation errors do not produce as severe consequences for head-tracked displays as they do for head-mounted dispays. However, it has been reported that head rotation errors cause the displayed objects to appear to distort and move slightly, and that tracking full head pose is still necessary to avoid having to keep the head in the correct orientation [56].

The following analysis demonstrates how head position and orientation errors can influence the perceived virtual world in head-tracked stereoscopic displays. The analysis is based on similar work for an augmented reality system using an optical see-through head-mounted display [68] and the evaluation of stereoscopic fusion control algorithms [180]. The two-dimensional parallel case is considered, because even this simple case shows the complicated relationship between head-tracking errors and the resulting distortion of the visual workspace. Figure 4.13 illustrates the effect of head-tracking errors in a simplified configuration. The vector between the actual eye positions **A** and **D** is parallel with the projection line. The modeled eye locations **B** and **C** are displaced due to head-tracking and viewing errors. Thus, the desired virtual location **E** is displaced to **F** resulting in a

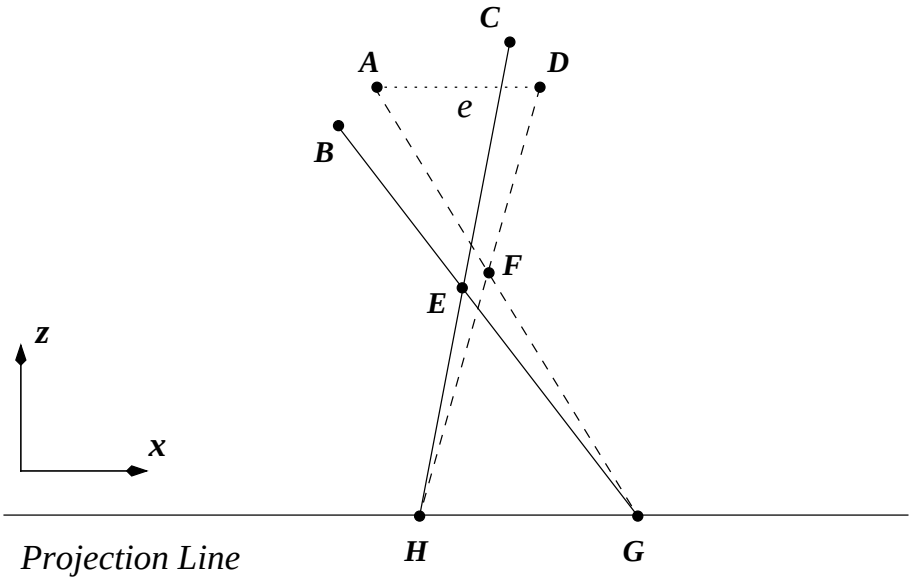


Figure 4.13. Parameters for the analysis of head-tracking errors. The actual eye separation vector is parallel with the projection line.

distortion of the visual workspace. The perceived location **F** is related to the parameters of the model by [68, 179]:

$$F_x = \frac{A_z D_x G_x - A_x D_z H_x + (D_z - A_z) G_x H_x}{A_z D_x - A_x D_z + D_z G_x - A_z H_x} \tag{4.16}$$

$$F_z = \frac{A_z D_z G_x - A_z D_z H_x}{A_z D_x - A_x D_z + D_z G_x - A_z H_x} \tag{4.17}$$

where:

$$H_x = \frac{(E_x - C_x) C_z}{C_z - E_z} + C_x \tag{4.18}$$

$$G_x = \frac{(E_x - B_x) B_z}{B_z - E_z} + B_x \tag{4.19}$$

The effect of position and orientation errors are considered separately. For position errors, the modeled and actual eye locations are related by:

$$\mathbf{B} = \mathbf{A} + \boldsymbol{\varepsilon} \tag{4.20}$$

$$\mathbf{C} = \mathbf{D} + \boldsymbol{\varepsilon} \tag{4.21}$$

where $\boldsymbol{\varepsilon}$ is the position error of the head tracker. Knowing that $A_z = D_z$, it follows from equation 4.17 that the magnitude of the error between the modeled and perceived locations is:

$$\|\mathbf{F} - \mathbf{E}\| = \frac{|E_z|}{|A_z + \varepsilon_z|} \|\varepsilon\| \tag{4.22}$$

Suppose that the tracking error is significantly smaller than the eye distance from the projection line and that the point on the object is between the eye separation vector and its reflection over the projection line:

$$|E_z| \leq \lambda \, |A_z| \tag{4.23}$$

$$|A_z| - |\varepsilon_z| \leq \kappa \, |A_z| \tag{4.24}$$

The perceived visual distortion is bounded by the tracking error:

$$\|\mathbf{F} - \mathbf{E}\| \leq \frac{\lambda}{\kappa} \|\boldsymbol{\varepsilon}\| \tag{4.25}$$

Typical values for the constants are $\lambda \leq 0.6$ and $\kappa \geq 0.9$. Thus, if the displayed objects are within the fusible stereoscopic range, they are distorted by no more than the head position error. Another consequence of equation 4.22 is that the distortion is a function of the object distance from the projection line, therefore the visual errors are smaller near

the display surface. Previous work arrived at the same conclusion [68]. The effect of head position errors is illustrated in Figure 4.14.

Head orientation errors result in a more complicated distortion of the perceived space. Considering the component of the orientation error along the z axis only:

$$B_x = A_x \tag{4.26}$$

$$B_z = A_z - \varepsilon \tag{4.27}$$

$$C_x = D_x \tag{4.28}$$

$$C_z = D_z + \varepsilon \tag{4.29}$$

the error components can be approximated by:

$$|F_x - E_x| \approx 2\frac{|E_z|}{|A_z|}\frac{|(A_z - E_z)(I_x - E_x) + (A_x - D_x)\varepsilon|}{|(A_z - E_z)(A_x - D_x) + 2(I_x - E_x)E_z\varepsilon|}\varepsilon \tag{4.30}$$

$$|F_z - E_z| \approx 2\frac{|E_z|}{|A_z|}\frac{|(A_x - E_x)(D_x - E_x)|}{|(A_z - E_z)(A_x - D_x) + 2(I_x - E_x)E_z\varepsilon|}\varepsilon \tag{4.31}$$

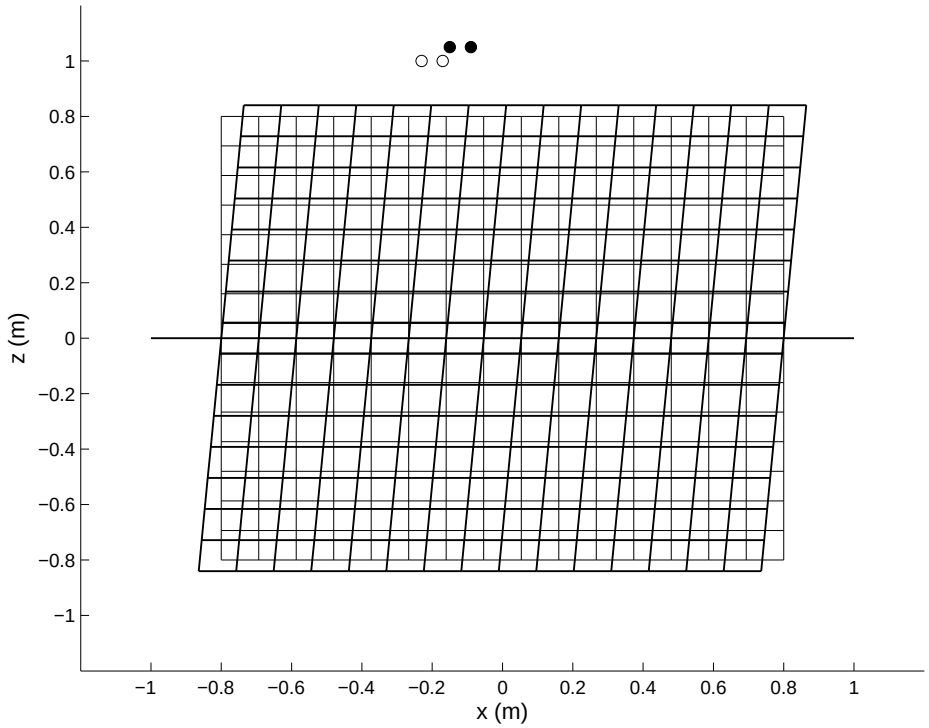


Figure 4.14. The effect of head position errors. The perceived virtual space appears scaled and sheared. The modeled and actual eye locations are represented by the empty and filled circles, respectively. The thin grid lines indicate the true workspace, the thick grid lines represent the one seen by the user.

where $\mathbf{I}$ is the midpoint between eyes:

$$\mathbf{I} = \frac{\mathbf{A} + \mathbf{D}}{2} \tag{4.32}$$

Two observations can be made from the error approximations. First, the error is proportional to the distance of the object from the projection line, in a similar way to position errors. Second, the error also grows as the object moves further away from the bisector between the eyepoints. Thus, the error is low within a diamond shaped region centered around the intersection of the bisector line between the eyes and the projection line, as illustrated in Figure 4.15. Notice that the perceived distortion grows rapidly outside this area. Fortunately, users typically focus straight ahead to the accurate region and cannot see objects within the degenerate region in front of the projection line. Thus, in general, the distortion caused by orientation errors is less noticable than the distortion due to position errors in head-tracked stereoscopic projection displays.

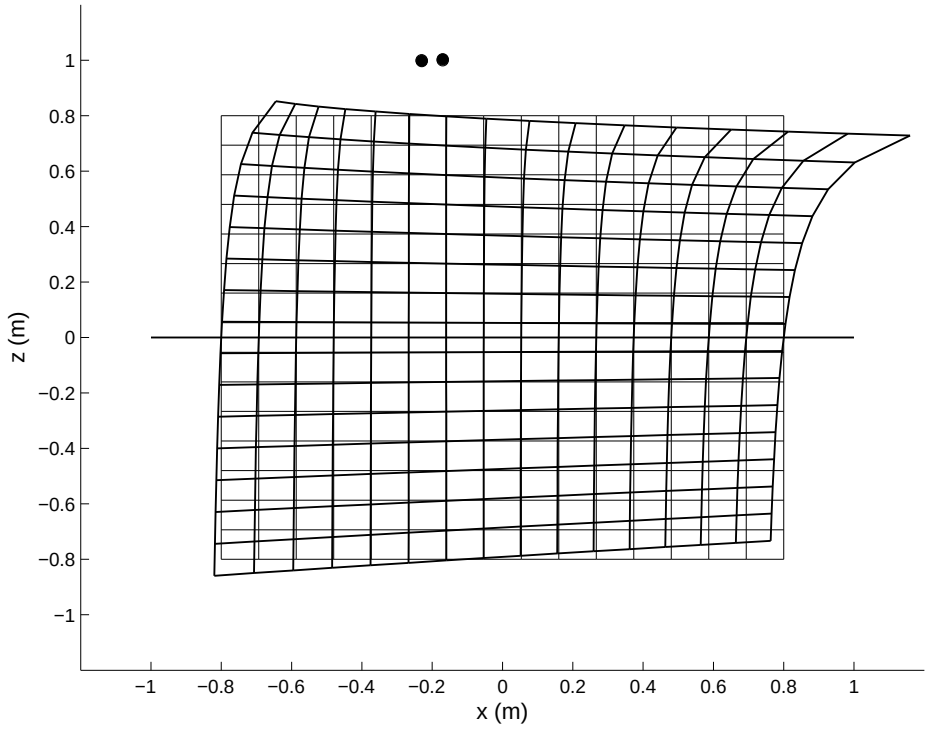


Figure 4.15. The effect of head orientation errors. Notice that the distortion is low within a diamond shaped region near the projection line.

4.6 Summary

In this chapter, several techniques were developed for improving and characterizing the display accuracy of combined visual and haptic displays. The methods reduce the static geometric error present in such systems by calibrating and colocating the tracking components. The methodology was evaluated using the Visual Haptic Workbench hardware, but it is also applicable to other similar configurations. For systems that provide an accurate head-tracking solution, the position tracker could be used for calibrating and registering the haptic interface. Finally, the remaining display parameters can either be measured directly or found indirectly by aligning real and virtual objects in the environment [53].

CHAPTER 5

VOLUME RENDERING STRATEGIES FOR INTERACTIVE DATA EXPLORATION

In this chapter, three techniques are presented that improve the interactivity of hardware-assisted volume rendering for typical data exploration tasks. All three methods increase rendering performance by reducing the number of fragments required to redraw the volume. The first two techniques, local rendering and image caching, are task-specific and can only be used with static displays, where the viewpoint does not change continuously or frequently during exploration. The third technique, focus and context decomposition, can be used to achieve real-time visual update rates for head-tracked stereoscopic displays, such as the Visual Haptic Workbench.

The techniques rely on the following features provided by current generation graphics hardware:

- **Render or copy to texture**: Saving the contents of the color and depth buffers is required for caching images and clipping geometry.
- **Programmable depth write**: Writing the depth buffer from a fragment program allows for arbitrary modifications to the contents of the buffer.
- **Early depth test**: The ability to discard fragments before they enter the fragment processing stage of the pipeline can be used to reduce the number of fragments required to render the volume. The early depth test provides an efficient method for achieving per-fragment control of what parts of the proxy geometry are drawn. Thus, it is not necessary to change the slice-based volume rendering algorithm, as long as a sufficient number of fragments are rejected during rendering.

5.1 Local Rendering

Volume rendered images are frequently combined with opaque geometry generated by visualization techniques and user interface components. Typically, the geometry covers only parts of the image. Thus, assuming a fixed viewpoint, it is not necessary to redraw the whole volume when the geometry changes as a result of user input. Local rendering combines a cached image of the volume with the result of rendering only the parts of the volume that are in front of the opaque objects, as illustrated in Figure 5.1. The algorithm consists of the following steps:

1. Clear the color and depth buffers. Draw the opaque objects into both and copy the depth buffer into a depth-component texture.

2. Draw a screen aligned rectangle with framebuffer blending turned on. In a fragment program, set the depth output to z_{near} for those fragments that have a corresponding z_{far} value in the depth texture obtained in the previous step. For the same fragments, output the color and opacity values from the image cache. For the remaining fragments, which will be updated in the next step, output the input depth value from the depth texture, as well as zero color and opacity for compositing.

3. Render the volume into the color buffer with the depth test enabled and depth write disabled.

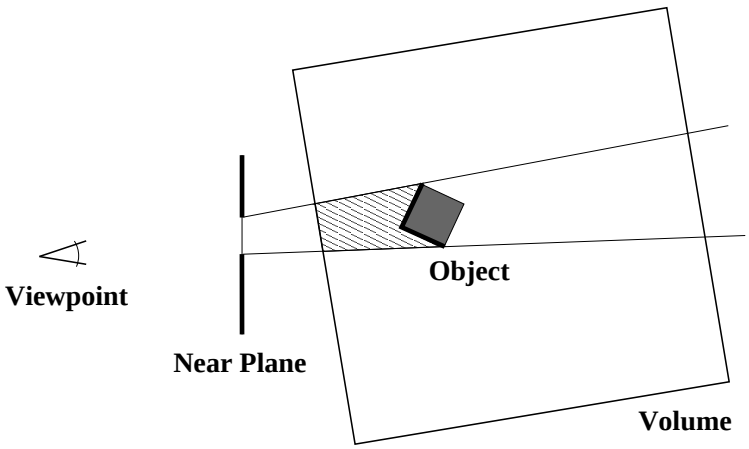


Figure 5.1. The concept of local rendering. Local rendering can be used to more efficiently combine opaque objects with a volume. Only the shaded region is redrawn, the remaining parts of the image are fetched from a cached copy. The depth buffer is set up as shown by the thick lines.

One example application of local updates is interactive placement of icons and stream constructs, such as streamlines, as shown in Figure 5.2. Local rendering can result in a significant speedup if only a small number of pixels are updated during each frame. In general, however, the degree of speedup varies depending on the screen coverage and depth range of the opaque objects in the scene.

The advantage of local updates is that the quality of rendering is not affected by the algorithm. For data probing tasks, local rendering offers a significant advantage as long as only a small region around the probe is updated. However, the algorithm assumes that the viewing parameters are fixed and the volume cannot rotate, limiting its utility to static displays and data probing tasks.

5.2 Image Caching

It is also possible to improve rendering speed by considering the effective range of the interaction technique. Image caching is based on subdividing the volume into blocks and saving rendered images for each block separately in a cache. Thus, during rendering, the cached images can be reused for those parts of the volume that are not influenced by the task.

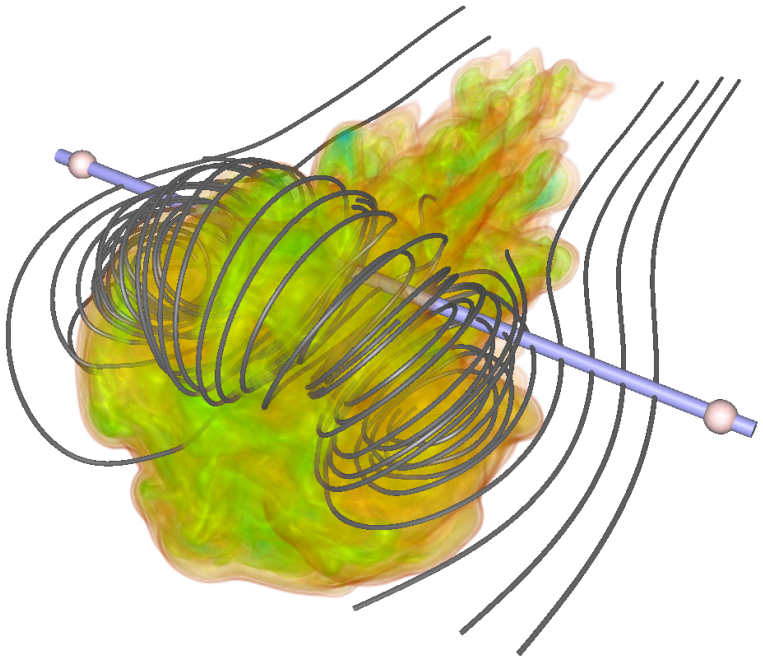


Figure 5.2. Example of local rendering. Interactive exploration of streamlines in vector field data is combined with a volume rendered view of the corresponding scalar data set.

One example where image caching can be applied is manipulating clipping planes or using them as frames of reference during exploration. As illustrated in Figure 5.3, if the volume is divided into slabs along the plane normal, only the slab that contains the plane has to be redrawn. The images for the remaining slabs are read from the cache and composited along the viewing direction.

The advantage of images caching is that it accelerates rendering of clipped volumes for simple clip geometries. The disadvantage is that extra memory is needed for storing the cached images and that the method works only for fixed viewing parameters and volume orientations. An example application of image caching is shown in Figure 5.4.

5.3 Focus and Context Decomposition

Even though the techniques presented in the previous two sections improve the interactivity of the system for selected visualization tasks, they do not provide a general solution to the problem of limited rendering resources. In particular, the methods do not guarantee a minimum update rate, which is essential in head-tracked immersive environments, where the whole volume has to be redrawn at every frame. In addition, typical navigation tasks, such as rotating, translating, and zooming the volume can be very cumbersome to perform, unless a reduced sampling rate is used during the operations.

Focus and context decomposition addresses the problem of limited rendering resources by restricting high-quality rendering to a region of interest in the volume and using a lower

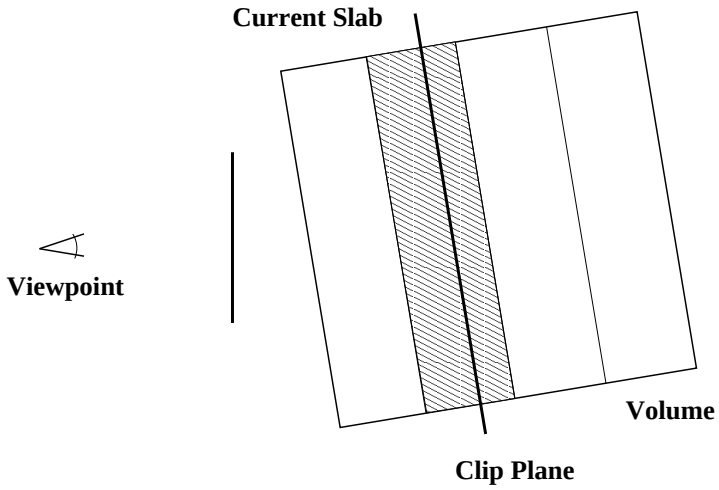


Figure 5.3. The concept of image caching. By decomposing the volume into slabs and caching images for each, only the current slab has to be redrawn as the location of the plane is changing in the volume.

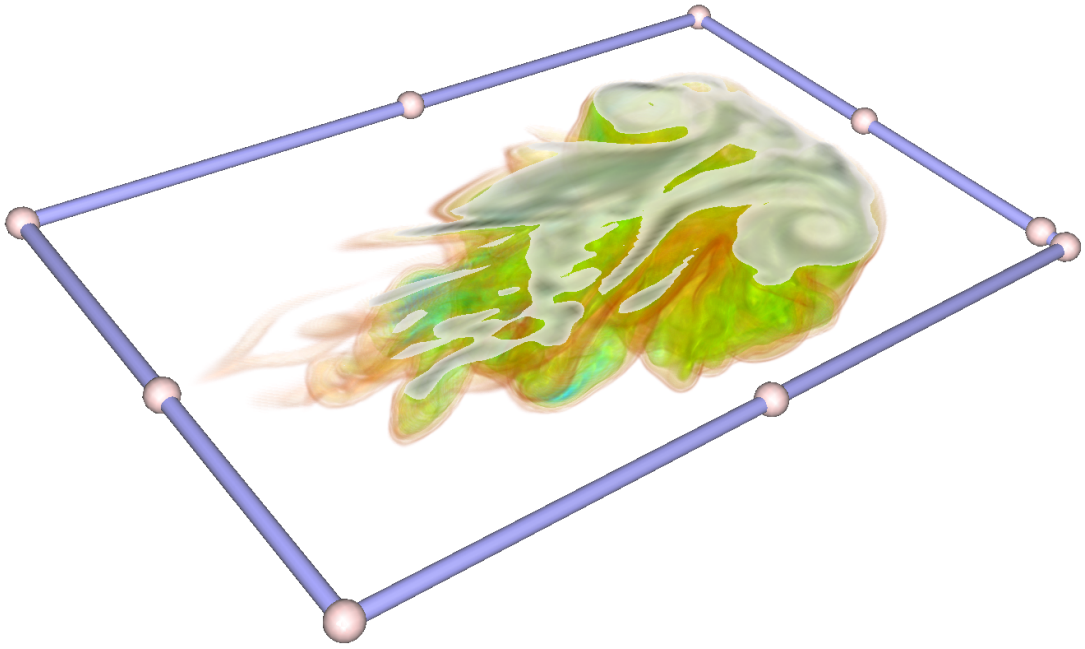


Figure 5.4. Example of image caching. Interactive exploration of a data set using axis aligned clipping planes can often reveal internal details hidden in the global view.

quality but faster technique for the remaining portions. In contrast to other existing acceleration techniques, such as early-ray termination and empty-space skipping [99], using a data-independent decomposition allows the user to explicitly balance the achievable quality and speed of rendering, independently of the selected transfer function. In addition to providing a way to control the speed of rendering, the focus region can also act as a 3D magic lens filter [174] showing more details or illustrating different aspects or features in the data.

There are three ways to add an interactive lens to a texture-based volume renderer:

- Tessellate the proxy geometry into focus and context polygons by slicing the clip object that represents the lens. The advantage of using this approach is that no redundant fragments are generated, because clipping is done completely on the CPU. The disadvantage is that computing the tessellation for multiple lenses or bricked volumes is slow and complicated.

- Tag the lens using a clip or segmentation volume. This approach is the most straightforward to implement, but requires an extra data lookup for each fragment. Also, depending on the resolution of the segmentation volume, changing or moving

the lens can be slow. A further restriction is that the sampling has to be the same for the focus and the context. Finally, trilinear interpolation of the segmentation volume is needed to avoid artifacts due to the finite resolution of the volumes.

- Render the volume in multiple passes and use depth-based clipping to cull away the unnecessary parts of the proxy geometry during rendering. This method avoids the complicated tessellation used by the first method by moving the clipping task to the GPU, but generates redundant fragments during rasterization. By exploiting the early depth test, however, the majority of the extra fragments can be quickly discarded before they enter the fragment processing stage of the pipeline.

The third approach is the most suitable for implementing multiresolution focus and context decomposition, because it permits the use of different sampling rates inside and outside the focus and supports arbitrary convex clip objects. In addition, depth-based clipping avoids the complicated tessellation algorithm required for the first technique.

To exploit the early depth test as much as possible and to incorporate preintegrated classification, explained in Section 5.4, the algorithm renders the volume in four passes, as illustrated in Figure 5.5. Note that volume clipping requires a second depth test, which is performed explicitly in the fragment shader. The steps of the algorithm are the following.

1. Clear the depth buffer to z_{far}. Render the front facing polygons of the clip object into the depth buffer and set the depth test to `GL_LESS`. Draw the proxy polygons for the context in front to back order. Make sure the depth test is enabled, but depth write is disabled when drawing the proxy geometry.

2. Render the back facing polygons of the clip object into a depth component clip texture. Reuse the contents of the depth buffer from the first step, but change the depth test to `GL_GREATER`. Draw slices for the focus between the front and the center of the clip object. Perform a second depth test at the beginning of the fragment program by reading the corresponding back face depth from the clip texture and terminating execution of the shader if the fragment depth is greater than the fetched value. The two depth tests together ensure that fragments are discarded as early as possible in the pipeline.

3. Render the front facing polygons of the clip object into the clip texture. Clear the depth buffer to z_{near}, draw the back facing polygons of the clip object into the

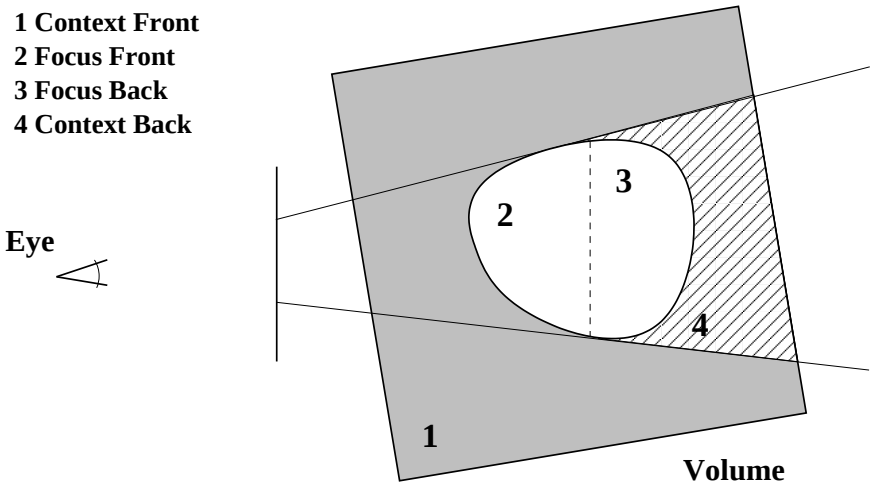


Figure 5.5. Rendering a volumetric lens using depth-based clipping. The numbered regions indicate how the volume is divided up by the steps of the algorithm.

depth buffer, and set the depth test to `GL_LESS`. Draw slices for the focus between the center and the back of the clip object. In the fragment shader perform a second depth test using the clip texture, similarly to the previous step.

4. Clear the depth buffer to z_{far}. Render the back faces of the clip object into the depth buffer and set the depth test to `GL_GREATER`. Draw slices for the context from the front of the clip object to the back of the volume.

The implementation is greatly simplified for screen space 2D lenses, because the focus does not need to be divided and the clip texture is not required. To incorporate opaque geometry into the rendering and to exploit the acceleration techniques of early ray termination and empty space skipping, two additional depth textures are needed: one to store the depth of opaque objects in the scene, and another to store the depth buffer for the current pass, which is used for initializing the depth buffer for the intermediate steps of the acceleration technique [99]. Also, additional passes may be needed to add a transition region for blending the results of sampling two different resolution versions of the data set at each slice. However, the transition region is often not necessary, it can make the visualization more confusing, and decreases rendering performance due to its nonconvex shape.

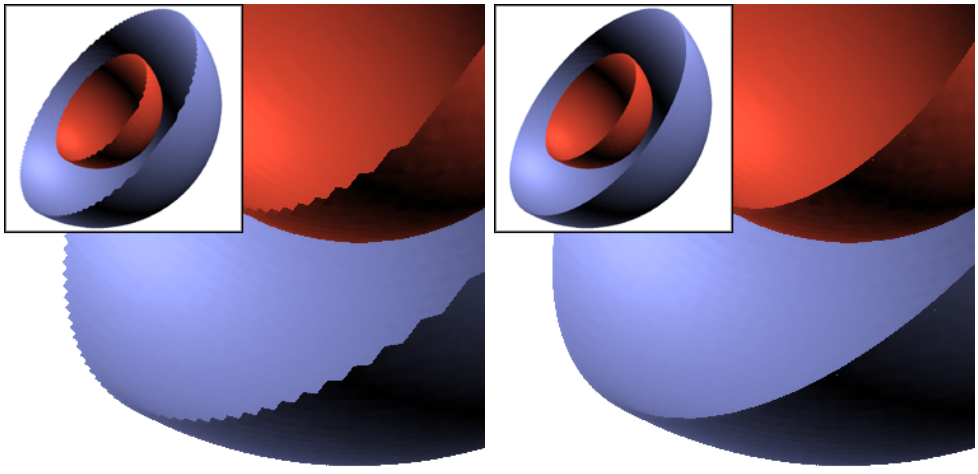
Even though the algorithm uses multiple passes to render the volume, only the required fragments pass through the shading and compositing stages, because the two depth tests discard the unnecessary fragments as early as possible in the pipeline. Dividing the focus

into front and back regions ensures that the majority of the fragments are rejected by the early depth test and do not enter the fragment processing stage at all. Splitting the focus is also necessary for the proper treatment of preintegrated clip boundaries, as described in the next section. Finally, clearing the depth buffer multiple times yields only minimal performance overhead because of the compressed multilevel depth representation used in current GPU architectures.

5.4 Preintegrated Clipping

The advantage of preintegrated classification is that it alleviates the aliasing problems that result from using an insufficient number of samples during rendering. As illustrated in Figure 5.6, clip boundaries need special treatment to avoid the artifacts caused by using a low sampling rate for achieving interactive update rates.

Two different methods have been used to handle the clipping problem in the past. The simpler solution is to map an opaque texture onto the clip geometry to hide the clipping artifacts. In most cases, however, the goal of clipping is to reveal the relationship of features, for example, the distances between isosurfaces. Thus, hiding the clipping artifacts also removes important details from the image. Recently, an optical model based on the combination of volume and surface shading was proposed that introduces a boundary layer of finite thickness around the clip object [183]. The problem with this



(a) (b)

Figure 5.6. Illustration of preintegrated clipping. Removing clipping artifacts when using preintegrated volume rendering with a low sampling rate. The images show a test data set (a) without and (b) with clip correction.

approach is that it requires interleaved rendering of the volume and the clip surface when using depth-based clipping, which results in a significant performance penalty.

To correctly handle clip boundaries, the clipping surface is drawn separately from the clipped volume. The contribution of the ray segments between the surface and the closest slice in the volume is computed explicitly in a fragment program. The computation is illustrated in Figure 5.7, which assumes that the clipped volume is located behind the clipping surface. First, the algorithm computes the intersection Q of the viewing ray that traverses point P on the surface with the first proxy polygon behind P. Next, data values v_P at P and v_Q at Q are fetched from the volume and the corresponding sampling distance d' is computed. For consistency, the algorithm does not use the actual length of the ray segment between P and Q, but the length of its projection along the viewing direction. The orthographic approximation of ray length is a commonly used technique that avoids the need for a 3D transfer function lookup table, because the same sampling distance can be assumed everywhere in the volume. The approximation usually does not cause serious rendering artifacts, because the sampling distance does not change rapidly across the volume. In contrast, at the clip boundaries, the sampling distance d' varies between 0 and d with abrupt changes across the slices (Figure 5.6). Thus, to correctly render the volume at the clip boundaries, a 3D lookup table is required, from which the

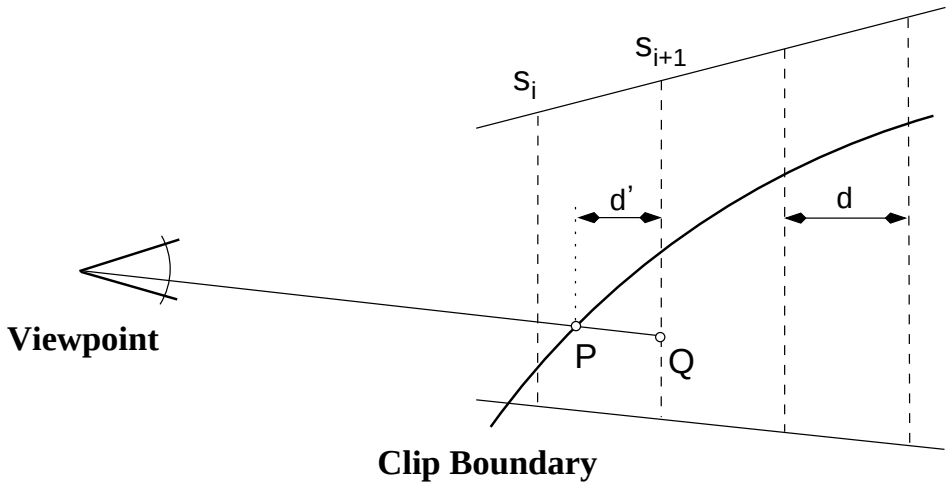


Figure 5.7. Preintegrated volume rendering at clip boundaries. When the clipped region is behind the clip boundary, the boundary is rendered before the volume. The contribution of the ray segment between the boundary and the first slice in the clipped volume is computed in a fragment program.

preintegrated color and opacity for scalar values v_P and v_Q and sampling distance d' is found. Note that the 3D table is needed for rendering only the boundaries of the clip geometry as opposed to the whole volume. Also, a small table usually suffices, because the size of the table along the length axis does not have to be large to yield an adequate approximation to the analytical preintegrated transfer function.

The method described above assumes that the front slices are projected to the back slices when computing the texture coordinates for the preintegrated data lookup. When the volume is in front of the clipping surface, which happens in steps 1 and 3 of the algorithm presented in Section 5.3, the surface is rendered after the volume, and the texture coordinates are computed by projecting points on the surface to the previous slice in the volume. In addition, the back slices are projected to the front slices for rendering the volume in these regions. To ensure that ray contributions for the slices in the center of the object are not included twice, a second depth test is added to the beginning of the fragment program to check whether the projected positions are not behind or in front of the corresponding point on the clip object.

5.5 Interactive Transfer Function Updates

A major drawback of preintegrated classification is that transfer function updates are computationally expensive. Naïve computation of the 3D transfer function lookup table by numerical integration is of $O(n^4)$ complexity [144]. Incremental preintegration reduces the complexity to $O(n^3)$ and significantly accelerates the construction of the table [181]. Previous implementations computed each slice of the table either by numerical integration on the GPU [144] or incrementally from previous slices on the CPU [181]. Both approaches limit the interactivity of transfer function editing, since each update takes a few seconds to compute. In the former case, numerical integration for each slice requires compositing a large number of screen aligned rectangles into a buffer, which is of the same complexity as volume rendering the whole table with a very high sampling rate. In contrast, the latter approach, although it requires far fewer operations, does not exploit the inherent parallelism of GPU computations. In addition, the lookup table from the CPU side has to be downloaded to GPU memory after every update, which can be an expensive operation that further reduces the attainable level of interactivity.

To achieve interactive updates, the two approaches are combined by computing the 3D lookup table using incremental preintegration on the GPU. Thus, the table is

immediately available for rendering after it is updated. The computation is divided into two parts. First, the base slice of the 3D table is constructed using a modified version of the algorithm presented in [144]. The difference is that the modified algorithm uses opacity-corrected associated colors instead of chromacities and front-to-back instead of back-to-front compositing. A floating-point pixel buffer is used to minimize round-off errors during computation. Linear interpolation of the colormap entries is performed explicitly in the fragment program, because current generation hardware supports only nearest neighbor interpolation for floating point texture lookups.

In the second stage, each remaining slice is incrementally constructed from the base slice and the previously computed slice. To maximize performance, two sub-buffers of a single pixel buffer are used in an alternating fashion. Before the current slice becomes the previous slice, its contents are copied to the corresponding location in the 3D lookup table texture. Note that the slices are computed at floating point accuracy, but are quantized to 8 or 16 bits during the copy.

Table 5.1 summarizes the performance measurement results for hardware-assisted incremental preintegration. The measurements were collected on a PC with two 2.0 GHz Pentium 4 CPUs, 2.0 GB system memory, a 4x AGP bus, and an ATI Radeon 9800 Pro graphics card with 256 MB video memory. Four samples per texel were used for the base slice computation. The results represent the time it takes to compute an 8-bit RGBA table from a 1D floating point transfer funtion texture. The table clearly shows that by moving the computation of the 3D preintegration table to the GPU, transfer function manipulation at interactive rates becomes possible.

Table 5.1. Timing results for updating the preintegrated transfer function table. The results are given in milliseconds.

Table Size	GPU			CPU		
	Base	Incr.	Total	Base	Incr.	Total
$128 \times 128 \times 8$	56 ms	3 ms	59 ms	1340 ms	81 ms	1421 ms
$128 \times 128 \times 128$	56 ms	29 ms	85 ms	1340 ms	1604 ms	2944 ms
$128 \times 128 \times 256$	56 ms	57 ms	113 ms	1340 ms	3214 ms	4554 ms

5.6 Bricking with Virtual Texture Coordinates

In addition to the fill rate limitation, volume rendering techniques are also limited by the fixed amount of texture memory available on current generation GPUs. In general, large data sets are rendered by subdividing the volume into a number of bricks, such that each brick fits into the available memory. The volume is rendered by downloading and rendering the bricks individually in sorted order towards the viewer. Typically, a fixed number of texture objects are allocated and get reused for subsequent bricks during rendering.

One application of interactive lenses is exploration of large data sets by rendering a high resolution magnified view of the data in the focus and a downsampled version in the context, as described in Section 5.7.1. Magnification lenses require on-the-fly texture coordinate modification, which causes problems at the boundaries when keeping the data in separate texture objects. To circumvent this problem, the bricks are allocated in a single memory pool together with an index volume, which is used for locating the bricks during data lookup, as shown in Figure 5.8. Adding a level of indirection forms the basis of the virtual texture memory scheme proposed recently [107]. In this case, however, the virtual to physical memory translation is slightly more complicated, because memory pages overlap at their boundaries. Assuming a single voxel overlap and an 8-bit index texture, the address computation comprises the following steps.

1. Compute index texture coordinate t_i from the virtual texture coordinate t_v, number of voxels n_v and number of bricks n_b along the axis:
$$t_i = \frac{t\, n_v - 0.5}{n_v - n_b} \tag{5.1}$$

2. Terminate execution of the shader if $t_i < 0$ or $t_i > 1$.

3. Look up the brick index i_b from the index texture and compute the physical brick address:
$$t_{pb} = i_b\, \frac{s_b\, 255}{n_p} \tag{5.2}$$
where s_b is the size of the brick and n_p is the number of voxels along the corresponding axis in physical memory space.

4. Compute the virtual offset t_{vo} within the brick by taking the fractional part of $t_i\, n_b$. The physical offset is obtained from:
$$t_{po} = \frac{t_{vo}\,(s_b - 1) + 0.5}{n_p} \tag{5.3}$$

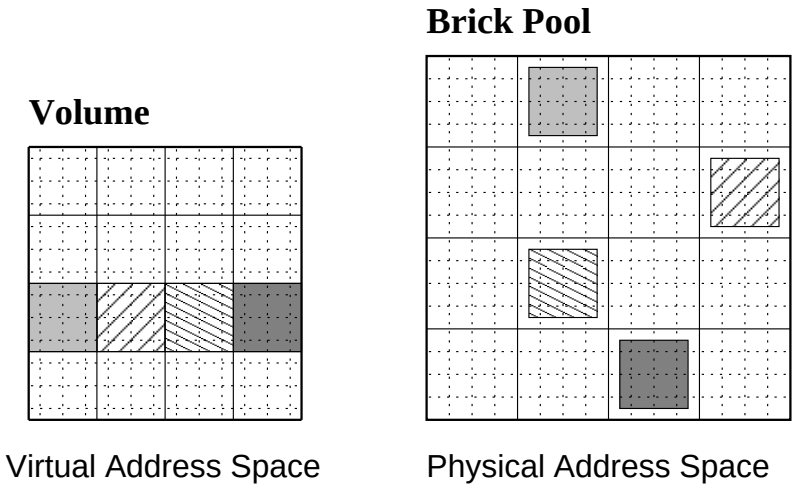


Figure 5.8. Using virtual texture coordinates for bricking. Keeping bricks in a single memory object and using virtual texture coordinates permits arbitrary coordinate transformations in the fragment shader.

5. Add t_{po} to t_{pb} to obtain the physical address t_p.

Virtual texture coordinates have several other applications. First, the addressing scheme makes preintegrated rendering of bricked data possible without having to clip the bricks at their boundaries using the method described in Section 5.4. Keeping a number of smaller bricks instead of a few large ones is also advantageous for out-of-core streaming implementations. Second, using virtual addressing, arbitrary texture coordinate transformations can be applied in the fragment shader, which allows for the implementation of various volumetric effects, such as volume deformation and perturbation methods.

The address translation incurs minimal performance overhead, because the index texture is small and the computation of the XYZ coordinates is done in parallel. When texture coordinates are not modified on the fly, the translation can further be optimized by adapting the slicing algorithm and computing the index texture coordinates on the CPU, thereby eliminating steps 1 and 2 of the computation.

5.7 Interactive Lenses for Volume Rendering

There are many different uses of interactive lenses in volume rendering applications. In the following subsections we consider three examples: detail in context viewing of large data sets, assigning different transfer functions to provide more flexibility for interactive exploration, and achieving considerable speedup through multiresolution rendering in time-critical applications.

5.7.1 Magnification

Volumetric magnifying lenses were proposed as a solution to the problem of viewing small details in large data sets together with their context [103]. To achieve a continuous transition between the magnified and normal views, a screen-space displacement texture is needed that modifies the texture coordinates before the volume data lookup in the shader. The displacements are scaled by the distance of the fragment from the viewpoint along the view direction to create a uniform screen-space magnification effect for each slice in the focus. In addition, to allow exploration of large data sets and to reduce aliasing problems, a high resolution version of the data is used in the focus and a downsampled version in the context. The virtual addressing scheme presented in section 5.6 allows the system to keep only the required portion of the high-resolution version of the data in texture memory and incrementally update it when the user moves the lens in the data.

Figure 5.9 shows an example of using the magnifying lens to inspect details in a turbulent fluid mixing data set. The magnification lens is useful for browsing very large data sets with limited resources, especially when only a small part of the original data fits into texture memory. In general, it is not possible to use traditional multiresolution strategies to show details in a similar way.

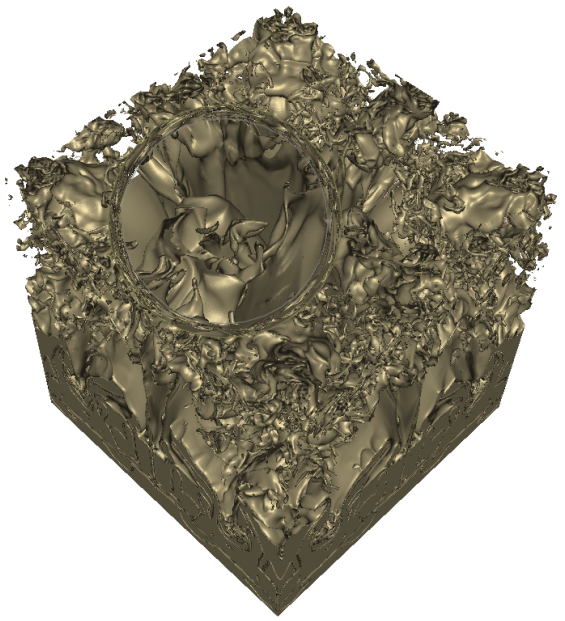


Figure 5.9. Example of using a magnification lens. The lens is used to view details in a turbulent fluid mixing simulation data set without losing the surrounding context.

5.7.2 Dual Transfer Functions

The goal of using transfer functions in volume visualization is to assign colors and opacities to features of interest in the data. Transfer functions can be specified in terms of direct or derived measures of the data values and locations. Typically, the transfer function is based on a single scalar value. It has been shown that transfer functions based on multiple measures are more successful at capturing and isolating features from each other [94]. One problem of using multidimensional transfer functions is the complexity of their design. It is very difficult to intuitively explore transfer function domains with more than three dimensions. Thus, volume rendering is frequently combined with other visualization techniques and interactive tools to aid the exploration process and to create more insightful illustrations.

Interactive lenses extend the existing set of exploration tools that are based on probing, clipping, and deforming the volume. By using different transfer functions in the focus and the context, it is possible to emphasize features that would be difficult to achieve with a single transfer function or with volume clipping alone. Screen space lenses are easier to manipulate than cutting and clipping objects and provide an alternative to switching transfer functions to show the external and internal details of objects, as illustrated in Figures 5.10 and 5.11(b).

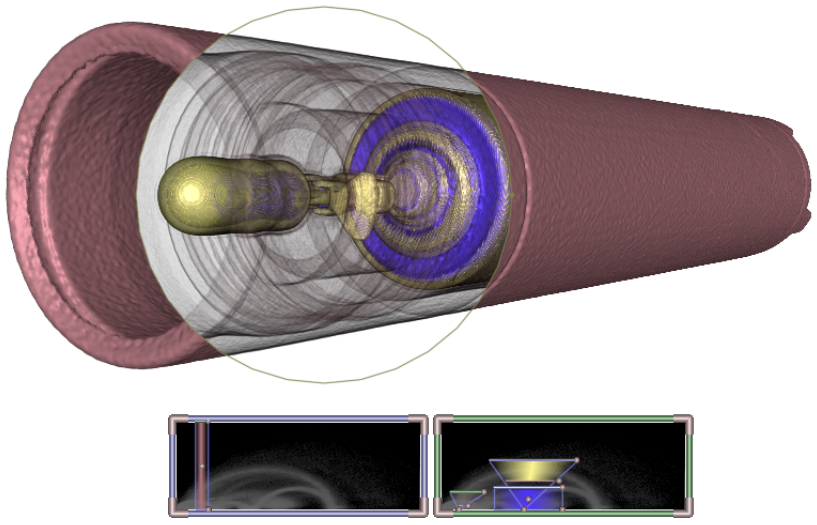


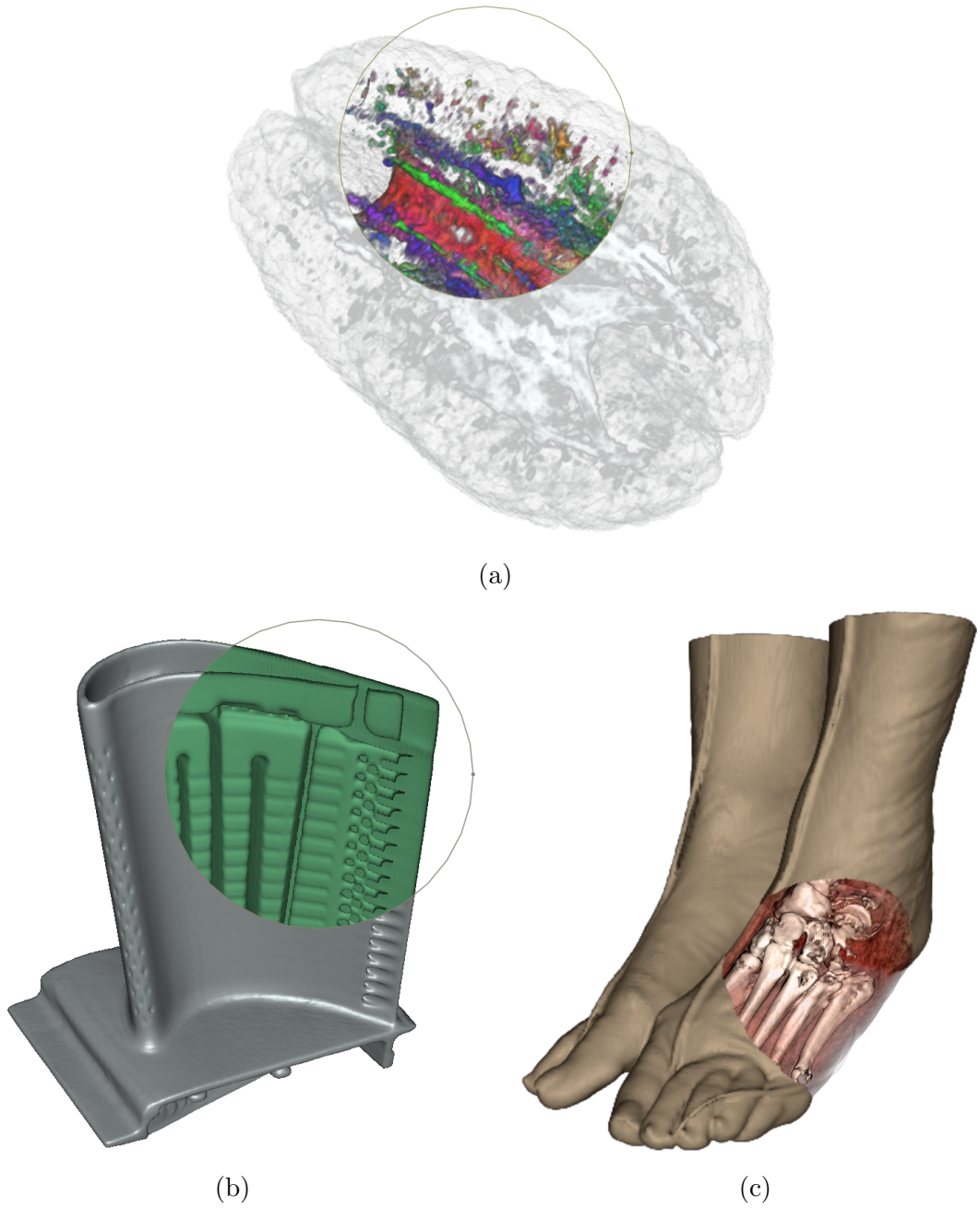
Figure 5.10. Example of using a transfer function lens. Different transfer functions are assigned to the focus and the context to show internal and external parts of a computed tomography scan of a flashlight.

Interactive lenses can also be used for visualization of different attributes in multidimensional data sets. An opaque detailed view in the focus within a transparent context is especially helpful in reducing clutter when interactively probing multidimensional data. One example is diffusion tensor field visualization, where the combination of a single measure in the context and multiple measures in the focus reduces occlusion, as shown in Figure 5.11(a). Interactive volume exploration, however, is best combined with multiresolution rendering to improve interactivity of the visualization system, as shown in Figure 5.11(c) and described in the next section.

5.7.3 Time-Critical Multiresolution Rendering

There are a number of factors that influence the performance of hardware-accelerated volume rendering techniques. Roughly speaking, performance depends on the number of fragments processed in the fragment pipeline, the amount of work each fragment requires, and the utilization of the texture and framebuffer memory caches. Multiresolution volume rendering techniques use subsampled data blocks to improve cache utilization and draw fewer slices to reduce the sampling rate. Additional ways to more efficiently balance the available processing capacity between the focus and the context include choosing a simpler and faster rendering algorithm for the context and reducing the size of the transfer function lookup tables and the viewport. Drawing the context into a smaller viewport and enlarging and compositing the resulting image is an effective way to reduce the number of fragments generated when zooming into the volume. In this case, the context is rendered in sufficient detail in the smaller viewport. A Gaussian filter can be used to reduce aliasing artifacts by slightly blurring the resulting image. Note that the goal of multiresolution focus and context rendering is not to create a seamless integration of the regions, but to achieve a better tradeoff between the quality and speed of rendering for improved interactivity in time-critical applications.

There are two cases for which the proposed decomposition works well. In the first case, a transparent context is used, mostly for orienting the focus in the data. The context does not have to be drawn at full resolution and can be sampled sparsely if the transfer function contains only low frequency components. The focus is attached to a data probe, showing multiple attributes together, possibly using multidimensional classification requiring high-quality rendering and complex transfer functions. The probe may also be combined with other visualization techniques, such as icons or streamlines,



(a)

(b)

(c)

Figure 5.11. Volume rendering examples using focus and context decomposition. (a) Volume rendering of a DT-MRI data set. The focus emphasizes the direction of the principal eigenvectors using an XYZ to RGB colormap. (b) View dependent illustration of external and internal isosurfaces in the turbine blade data set based on gradient directions. (c) A surgical planning scenario using the feet data set and a spherical focus region.

to provide further local information about the data. This kind of decomposition is useful for reducing clutter when exploring multidimensional or multifield data sets. In the second case, the context is opaque and provides an external view of an object, into which the focus is embedded for examining internal details, as shown in Figure 5.11(c). The latter scheme can efficiently exploit acceleration techniques for the context and permits high-quality detailed rendering for the focus.

5.7.3.1 Performance Evaluation

To obtain a quantitative assessment of the tradeoff between quality and performance, an evaluation was conducted consisting of rendering the same volume from the same view while varying the size of the focus, as illustrated in Figure 5.12. The experimental platform was a PC with two 2.0 GHz Pentium Core 2 CPUs, 2.0 GB system memory, and an ATI Radeon X1900 GT graphics card with 256 MB video memory. The fixed parameters for the evaluation are collected in Table 5.2. All textures had 8-bit components. A single viewpoint was chosen instead of a camera path because of the high variability of the performance of slicing a 3D texture from different directions on ATI cards. Thus, the

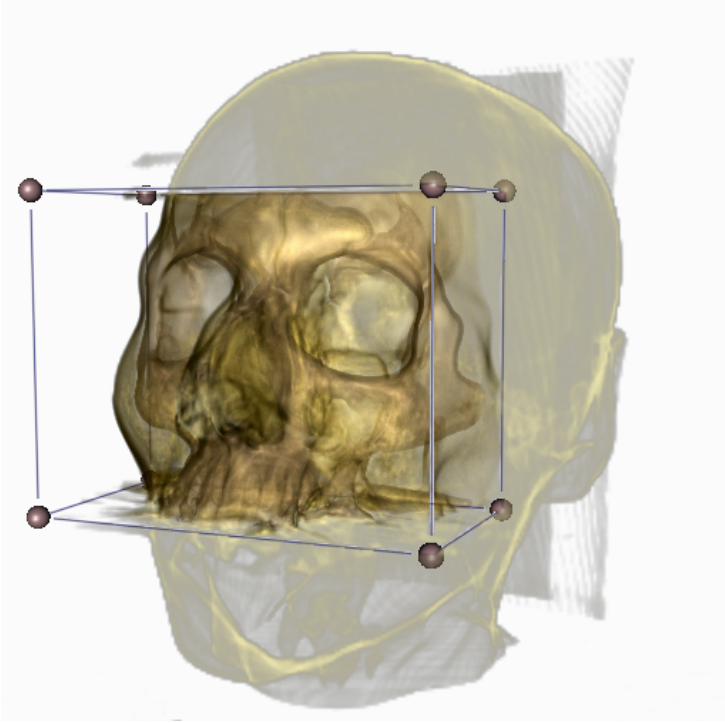


Figure 5.12. Volume rendering of the head data set used for the performance evaluation.

Table 5.2. Parameters for the performance evaluation of the method.

	Focus	Context
Volume Size	4×256^3	1×128^3
Lookup Table Size	5×256^2	4×128^2
Rendering Model	Phong Shading	Unshaded
Sampling Rate	4.0	1.0
Viewport Size	100%	50%

reported results reflect peak performance of the renderer. The acceleration techniques of early ray termination and empty space skipping were not used to obtain results that are the same for any data set and transfer function combination when rendered under the same conditions. The focus was placed always closest to the user, mimicking typical use of the interface.

The results of the experiment for different focus and viewport sizes are collected into Table 5.3. Focus size is represented as percentage of the whole data set size along a single axis and also in volume. The results in the first row correspond to an implementation without the overhead of the decomposition, which explains the small difference in performance when reducing the volume of the focus by half. The overhead mostly comes from setting up and using the clip textures and the depth buffer during each pass. The results show that by using focus and context decomposition, it is possible to achieve a 2-8 times speedup relative to the single high-resolution mode only. These results correspond to empirical observations on the interactivity of the system.

Table 5.3. Volume rendering performance using focus and context decomposition. The results are reported in frames per second and tabulated as a function of focus and viewport size and volume.

Focus Size		Viewport Size	
Axis	Volume	512^2	1024^2
100%	100%	8.45	3.60
79%	50%	11.52	4.48
63%	25%	20.28	7.76
50%	12.5%	35.71	13.19
40%	6.25%	63.29	20.94

5.8 Summary

In this chapter, three techniques were presented for improving the interactivity of hardware-assisted volume rendering in data exploration tasks. All three methods attempt to reduce the amount of fragments needed to redraw the volume. While the first two methods are task-specific, the third approach provides a generic solution for balancing the limited amount of resources available in typical volume rendering applications. The strength of the approach is that the attainable speedup does not depend on the transfer functions, only the rendering parameters used. Additional speedup is made possible by incorporating early ray termination and empty space skipping acceleration into the rendering algorihm. However, the transfer functions have to be chosen carefully for the region in which the acceleration technique is used, otherwise the rendering overhead can result in a loss of performance compared to using no acceleration at all. In theory, the system could detect when the user switches to using transfer functions for which the overhead of the acceleration yields slower performance and adjust the rendering mode accordingly. The results of the performance evaluation further demonstrate that by more efficiently balancing the available rendering resources, it becomes possible to incorporate volume rendering methods into immersive data exploration applications.

CHAPTER 6

A CONSTRAINT-BASED TECHNIQUE FOR HAPTIC VOLUME EXPLORATION

This chapter introduces constraints as a foundation for developing point-based haptic rendering algorithms for scientific data exploration. In their simplest form, volumetric constraints are constructed by augmenting the proxy point with local reference directions and controlling the motion of the proxy according to a set of rules and parameters along the reference directions. The constraint-based approach has the advantage that it provides a basis for developing haptic effects for a variety of data modalities. Hence, similar or closely related rendering methods can be applied to seemingly unrelated data sets, resulting in a familiar and consistent interaction experience.

6.1 Motivations and Requirements

The use of haptic constraints for data exploration tasks is motivated by several considerations. Many visualization methods use directional and shape cues to convey information about the properties of the data. As discussed in Chapter 2 Section 2.3.2, the corresponding haptic cues would be difficult to produce using static force fields. In contrast to force fields, constraints generate haptic cues that are suitable for guiding the user in the data. In addition, constrained haptic feedback plays an important role in everyday tasks, such as writing on a piece of paper or assembling objects. Thus, one can easily understand haptic effects that are related to real-world experience. As shown in Section 6.4, controlling the motion of the proxy makes the implementation of a variety volumetric effects possible. Finally, haptic user interfaces can also utilize constraints for precise placement and manipulation of user interface components.

Constraints serve a dual purpose in data exploration applications. First, constraints reduce the complexity of the domain of exploration by providing support and guidance for a particular task. Second, constraints are used as indicators to convey additional

information about the task or the data. A particular haptic mode cannot be effective unless guidance and indication are separated. The approach developed in this chapter is suitable for developing compound effects for simultanous guidance and indication. However, as shown in Section 6.5, point-based feedback is limited at creating volumetric effects without introducing ambiguities in the force output.

There are three general requirements that any haptic data rendering technique must satisfy. First, to provide the user with a consistent experience, the displayed haptic cues cannot result in conflicting visual feedback. For example, when exploring streamlines in vector field data, the path traversed by the haptic probe has to match the visual representation of the selected streamline. Second, the technique should provide the user with a way to freely explore the available workspace. This can be achieved by changing the feedback mode based on user input, indicated either through a separate input device or the force from the haptic interface. Third, in addition to creating strong haptic cues for guidance, the system has to provide visual or haptic indication whenever the user leaves a selected feature of interest in the data.

6.2 Haptic Rendering with a Constrained Proxy Point

In general, haptic data rendering algorithms based on a proxy point are built upon the following components that are executed during every iteration of the servo loop. See also Figure 6.1. As discussed in Section 6.6, with the exception of force computation, the components may be executed several times in a haptic frame as part of an iterative procedure.

1. Locate the probe and the proxy in the data.

   The point-location problem has a trivial solution for structured uniform grids, since the corresponding cell can be directly found using the affine transformation between the data space and the haptic workspace. For curvilinear and unstructured grids, point-location algorithms developed for particle advection can be readily applied [130]. Incremental point tracking also works outside the mesh as long as the boundary is convex and does not have holes. Otherwise, the algorithm may get stuck in local minima. In addition, multiple copies of state variables, including the current cell index and the Jacobian matrix, are needed to efficiently track multiple points in the data.

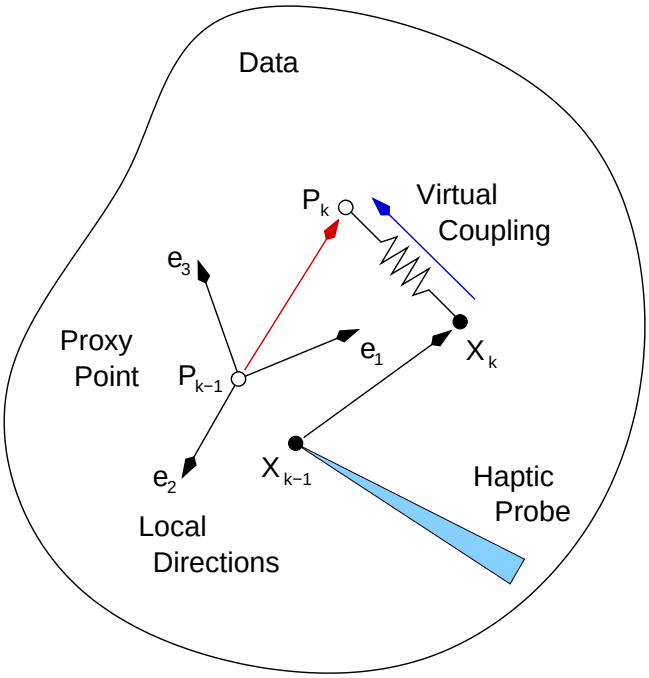


Figure 6.1. Components of constrained point-based haptic data rendering. At time step k the position of the haptic probe has changed from $\mathbf{x}_{k-1}$ to $\mathbf{x}_k$. The position of the proxy is updated from $\mathbf{p}_{k-1}$ to $\mathbf{p}_k$, from which the force response is computed using a spring coupler. Proxy update is based on data measures at the previous proxy location, motion rules, and rendering parameters.

2. Compute local data measures at the probe and proxy locations.

   Data values and other measures, such as gradient or curvature information are obtained from interpolating data elements around the current proxy and probe locations. The typical methods of choice are linear and trilinear interpolation, although higher order methods may be more appropriate depending on the scale and resolution of the display [152]. A set of local direction vectors $(\mathbf{e}_1 \ldots \mathbf{e}_n), n = 1, 2, 3$ is a key component of constructing haptic effects based on volumetric constraints. Examples include the vectors defined by the gradient and principal curvature directions in scalar data and the eigenvectors in diffusion tensor data. Sometimes the reference vectors may be ill-defined or may not exist. Thus, an important requirement from the haptic rendering algorithm is to compute a stable force response even when transitioning into and out of homogeneous and ill-posed regions in the data. For example, in scalar volumes the reference vectors are poorly defined

in regions where the gradient vanishes. One way to achieve smooth transitioning is to modulate the force output as a function of gradient magnitude [114]. Another example is the problem of specifying transfer functions such that isotropic regions are handled properly in diffusion tensor data. In this case the transfer function has to be constructed such that the force output either vanishes or the constraint degenerates to an isotropic point constraint.

3. Move the proxy to a new location in the data.

   The goal of the proxy update step is to create haptic effects that provide guidance to the user or emphasize features in the data. The motion of the proxy is specified by rules along the reference directions. In general, the actions of several motion rules are combined such that the resulting force response corresponds to the desired haptic effect, such as a penetrable barrier or viscosity effect. The individual contributions of the motion rules can further be modified via haptic transfer functions. For example, a transfer function can be used to specify apparent stiffness and friction for isosurface regions based on data value and gradient magnitude [114]. In contrast to visual transfer functions, haptic transfer function design is an unexplored area. Although it has been demonstrated that it is possible to faithfully reproduce measured material properties [133], synthesizing them from scientific data sets remains a difficult problem. This work uses stiffness and drag threshold transfer functions $\boldsymbol{\kappa}$ and $\boldsymbol{\tau}$ to constrain the motion of the proxy along the reference directions in the data.

4. Compute the force response.

   In general, the reaction force is computed from the current probe and proxy locations. The spring-damper form of virtual coupling expresses the force response as

$$\mathbf{F}_k = k_c \left(\mathbf{x}_k - \mathbf{p}_k\right) - b_c \left(\dot{\mathbf{x}}_k - \dot{\mathbf{p}}_k\right) \tag{6.1}$$

   where the stiffness and damping parameters k_c and b_c are chosen according to the device capabilities. The optimal choice maximizes the coupling stiffness without causing instability [2]. One problem is that these parameters may not be constant throughout the workspace. A particular choice that works well in the center of the workspace may cause instability near the perimeter. Nevertheless, the parameters can be tuned using a point constraint at different locations in the workspace and

determining which settings cause the device to become unstable by itself. In this work, the damping term is excluded from Equation 6.1, because filtering velocity is difficult without high resolution position measurements [32].

The following sections develop a general strategy for updating the location of the proxy point. First, the linearized approach is discussed. Next, one-dimensional motion rules are introduced based on the linear formulation. After demonstrating how transfer functions can be used for modulating the parameters of the rules according to local properties of the data, problems of the linearized approach are revealed. Nonlinear constraints are proposed as a solution to the problems of the linearized approach. Finally, the proxy chain algorithm is introduced as a general solution to combining multiple nonlinear constraints in haptic programming environments.

6.3 Linearized Formulation

The linearized approach assumes that each constraint is represented by a possibly incomplete orthonormal frame. Hence, the proxy can be updated along each axis of the frame independently,

$$\mathbf{p}_k = \mathbf{p}_{k-1} + \sum_{i=1}^{n} \mathbf{e}_i \, \Delta p_i \tag{6.2}$$

where the components of $\Delta\mathbf{p}$ are functions of the position of the probe relative to the previous location of the proxy. Thus, the components are expressed in the reference frame as

$$\Delta p_i = f(\Delta x_i) \tag{6.3}$$

$$\Delta x_i = \mathbf{e}_i \cdot (\mathbf{x}_k - \mathbf{p}_{k-1}) \tag{6.4}$$

By decomposing $\Delta\mathbf{x}$ along the reference axes, a variety of volumetric effects can be implemented. For example, the proxy can be locally constrained to a point, a line, or a plane, depending on how many axes it is restricted to move along, as shown in Figure 6.2. Compound effects are constructed by specifying motion rules and rendering parameters, as described in the following two sections.

6.4 Motion Rules

A motion rule is a one-dimensional function that implements a particular haptic effect. For example, to simulate viscous drag and plastic material behavior, the drag rule displays increasing resistance between the probe and the proxy until a certain threshold is reached.

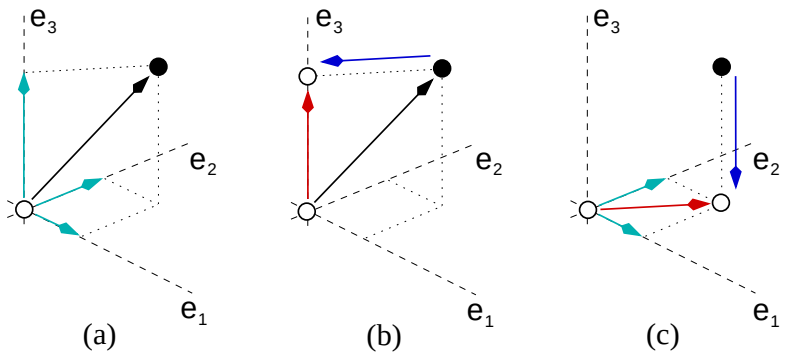


Figure 6.2. Linear proxy update along orthogonal directions. (a) By decomposing the proxy-probe displacement in the orthonormal frame formed by the direction vectors, a variety of effects can be expressed. For example, proxy motion can be (b) restricted to a single direction to express a line constraint or (c) restricted to two directions to express a planar constraint.

At this point, the proxy is allowed to move towards the probe, keeping the reaction force at the same level. The drag rule is expressed succinctly by the following formula

$$\Delta p_i = \text{sgn}(\Delta x_i)\ \max\left(|\Delta x_i| - \tau_i, 0\right) \tag{6.5}$$

Note that Equation 6.5 yields unconstrained motion when $\tau_i = 0$

$$\Delta p_i = \Delta x_i \tag{6.6}$$

Otherwise, a bilateral constraint is obtained, because the rule is symmetric about the origin. Depending on the magnitude of τ_i, either a viscous drag effect or a plastic deformation effect is created by the drag rule. Note, that the rule results in a stationary constraint when τ_i is sufficiently large, since it would take considerable effort to move the probe away from the proxy while resisting the increasing amount of attracting force between them. In this work, the term drag threshold is used for τ_i, since it controls the difficulty of dragging the proxy along a particular direction $\mathbf{e}_i$.

A unilateral constraint, which forms the basis for surface rendering algorithms, is created by considering the direction of travel along the reference axis. Assuming unconstrained motion is permitted on the positive side of the axis, the rule is expressed by the following set of equations

$$\Delta p_i = \begin{cases} \Delta x_i & \text{if} \quad \Delta x_i > 0 \\ \min\left(\Delta x_i + \tau_i, 0\right) & \text{if} \quad \Delta x_i \leq 0 \end{cases} \tag{6.7}$$

The problem with the drag rule is that it does not provide sufficient indication of when the proxy gets released from the fixed location. For certain applications, updating the proxy in discrete steps yields more informative feedback. The snap-drag rule implements such effect by moving the proxy closer to the probe as soon as the distance between them grows larger than τ_i

$$\Delta p_i = \begin{cases} \operatorname{sgn}(\Delta x_i)\,\tau_i & \text{if} \quad |\Delta x_i| > \tau_i \\ 0 & \text{if} \quad |\Delta x_i| \leq \tau_i \end{cases} \tag{6.8}$$

The two rules and the resulting force profiles are illustrated in Figure 6.3.

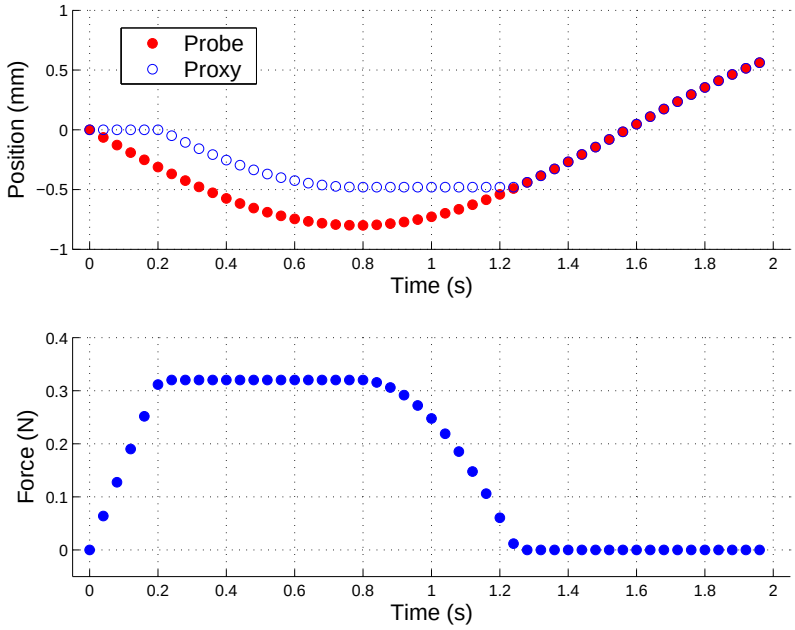
6.5 Transfer Functions

Transfer functions modulate the output of the motion rules by specifying the parameters of the rules. For example, the drag threshold τ_i can be used to blend between unconstrained motion, viscous drag, and plastic deformation. It is also possible to control the motion of the proxy by scaling the force output according to stiffness transfer function $\boldsymbol{\kappa}$,

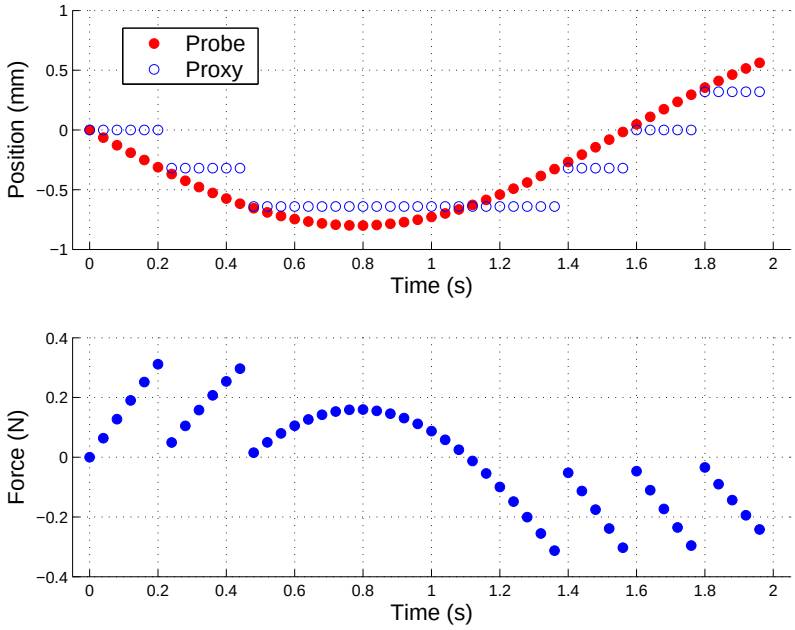
$$F_{k,i} = \kappa_i\, k_c\, (x_{k,i} - p_{k,i}) = k_i\, (x_{k,i} - p_{k,i}) \tag{6.9}$$

where $0 \leq \kappa_i \leq 1$. The stiffness transfer function controls the force required for dragging the proxy. Note that setting either τ_i or κ_i to zero produces no force output and creates frictionless motion along the axis. However, two different effects are obtained, because in the first case the proxy follows the probe exactly, while in the second case it lags behind by distance τ_i. However, both parameters are necessary for expressing a range of effects from subtle directional hints to stiff rigid constraints. To illustrate why one parameter cannot be substituted for the other, consider following an elastic surface with varying stiffness. As shown in Figure 6.4, changing the stiffness of the coupler is equivalent to moving a copy of the proxy towards the probe [149]. It is not possible to achieve the same effect with the drag constraint alone, since the probe would have to move to the other side of the proxy to drag it back to the starting position. Hence, it is necessary to maintain two separate but not completely independent parameters to create both elastic and plastic material behavior.

Little is known about finding the proper transfer function for a specific exploration task. Typically, to reduce ambiguities in the display, one strong major constraint is combined with several weak minor constraints. However, due to the limitations of point-based feedback, the effects of individual constraints cannot be completely separated



(a) Unilateral drag constraint.



(b) Snap-drag constraint.

Figure 6.3. Illustration of one-dimensional motion rules: (a) unilateral drag and (b) bilateral snap-drag. The motion of the probe and the proxy as a function of time is represented by the filled and empty circles, respectively. The resulting force responses are shown in the lower parts of the figures. Note that the sampling does not correspond to the haptic update rate.

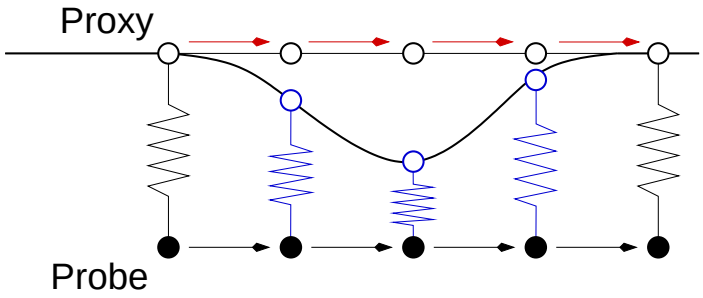


Figure 6.4. Example of haptic rendering parameters. Changing the coupling stiffness or moving a copy of the proxy for force calculations are equivalent for achieving varying elastic constraints. In this example the stiffness is reduced as the proxy follows the probe resulting in a "flexible" constraint.

from each other. To illustrate the ambiguity, consider two constraints along orthogonal directions $\mathbf{e}_1$ and $\mathbf{e}_2$. The net force resulting from the combination of the two constraints is

$$\mathbf{F} = \mathbf{F}_1 + \mathbf{F}_2 = k_1(x_1 - p_1)\mathbf{e}_1 + k_2(x_2 - p_2)\mathbf{e}_2 \tag{6.10}$$

The perceived stiffness along direction $\mathbf{e}_2$ is

$$\frac{\partial \|\mathbf{F}\|}{\partial \mathbf{e}_2} = \frac{\|\mathbf{F_2}\|}{\|\mathbf{F}\|} k_2 \tag{6.11}$$

Equation 6.11 shows that the perceived stiffness decreases as the magnitude of the total force increases. Thus, changing the stiffness parameter along one direction influences the perceived stiffness along the other direction. Friction algorithms compensate for the ambiguity by modulating the magnitude of the friction force with the penetration distance along the normal direction [126].

6.6 Nonlinear Constraints

The linearized formulation presented in Section 6.3 is suitable for creating haptic effects for variety of data modalities. Using the linear form is appropriate when the corresponding visual feedback is also built on the same representation. However, visualization algorithms typically define features of interest using implicit expressions, resulting in constraints that are nonlinear functions of position. Hence, the linear model can result in misleading visual and haptic feedback, as illustrated below.

Consider exploration of streamlines in vector field data using the follow haptic mode and the linearized formulation [39]. As illustrated in Figure 6.5, due to the poor numerical

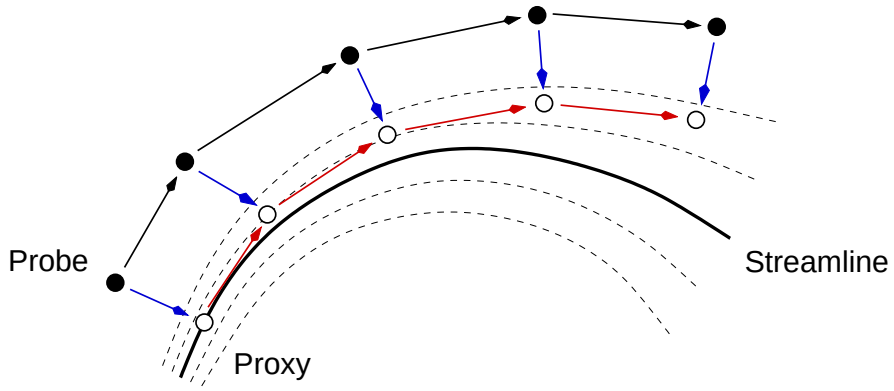


Figure 6.5. Exploring a streamline using the follow mode and haptic line primitives. Due to the poor numerical accuracy of the linear approximation, the proxy can drift from the selected streamline, resulting in conflicting visual cues. The error is most noticeable in regions of high field divergence.

accuracy of the linear approximation, the proxy can drift from the selected streamline, resulting in conflicting visual cues. The error depends on the divergence and vorticity of the vector field and the speed of exploration. The linear model is equivalent to using an Euler integrator for solving the ODE that describes the particle advection process,

$$\mathbf{p}_k = \mathbf{p}_{k-1} + \mathbf{v}(\mathbf{p}_{k-1})\,\Delta t \tag{6.12}$$

where $\mathbf{p}_k$ is the location of the proxy at time step k, $\mathbf{v}(\mathbf{p})$ is the field vector at position $\mathbf{p}$, and Δt is a function of $\mathbf{x}_k$, the position of the probe at time step k, relative to $\mathbf{p}_{k-1}$, the position of the proxy at time step $k-1$.

$$\Delta t = \frac{\mathbf{v}(\mathbf{p}_{k-1}) \cdot [\mathbf{x}_k - \mathbf{p}_{k-1}]}{\|\mathbf{v}(\mathbf{p}_{k-1})\|^2} \tag{6.13}$$

In practice, Euler integration is avoided for graphical display because of its poor numerical accuracy. Higher order schemes with adaptive stepsize control are preferred instead [139]. Hence, assuming that a function $\mathbf{f}_V(\mathbf{p}, \Delta t)$ is provided, which for vector field V solves the particle advection problem from starting position $\mathbf{p}$ and step size Δt, Equation 6.12 can be substituted with an iterative procedure,

$$\mathbf{p}_{k,j+1} = \mathbf{p}_{k,j} + \mathbf{f}_V(\mathbf{p}_{k,j}, \frac{\mathbf{v}(\mathbf{p}_{k,j}) \cdot [\mathbf{x}_k - \mathbf{p}_{k,j}]}{\|\mathbf{v}(\mathbf{p}_{k,j})\|^2}) \tag{6.14}$$

where $\mathbf{p}_{k,0} = \mathbf{p}_{k-1}$. The iteration terminates when the residual vanishes, which corresponds to a local extremum of the probe-streamline distance function. The advection

function is defined reversible.

$$\mathbf{f}_V(\mathbf{p}, -\Delta t) = \mathbf{f}_{-V}(\mathbf{p}, \Delta t) \tag{6.15}$$

If the iteration fails to converge, which may happen for example at critical points in the field, it falls back either to the linear approximation or, if detected, the location of the critical point. The experiment described in Chapter 7 uses a third-order Runge-Kutta method with fourth-order adaptive stepsize control as the advection function $\mathbf{f}_V$.

A similar iterative method can be used for constraining the motion of the proxy to an isosurface of a scalar quantity [150, 77]. To ensure that the proxy remains on the surface, the linear estimate is refined using Newton-Raphson iteration along the gradient direction,

$$\mathbf{p}_{k,j+1} = \mathbf{p}_{k,j} - \frac{\nabla s\left(\mathbf{p}_{k,j}\right)\left[s(\mathbf{p}_{k,j}) - s_0\right]}{\| \nabla s\left(\mathbf{p}_{k,j}\right) \|^2} \tag{6.16}$$

where $s(\mathbf{p})$ and $\nabla s(\mathbf{p})$ are the field value and gradient at position $\mathbf{p}$, respectively. The linear estimate is obtained by projecting the probe to the tangent plane at the proxy.

$$\mathbf{p}_{k,0} = \mathbf{x}_k - \nabla s\left(\mathbf{p}_{k-1}\right)\frac{\nabla s\left(\mathbf{p}_{k-1}\right) \cdot \left[\mathbf{x}_k - \mathbf{p}_{k-1}\right]}{\|\nabla s\left(\mathbf{p}_{k-1}\right)\|^2} \tag{6.17}$$

The refinement is terminated when the residual $[s(\mathbf{p}) - s_0]$ is sufficiently small. The algorithm falls back to the linear approximation when the refinement does not converge before reaching the maximum number of iterations permitted.

Volumetric constraints are augmented with a state variable indicating whether the constraint is active or not. For example, isosurfaces of interest may be specified by a set of selected isovalues. In this case, the algorithm distinguishes between a free motion state and a constrained state. Transitioning from the free state to the constrained state happens either when the probe crosses a surface with an isovalue from the set or when the probe is within a specified snap distance from the surface.

6.7 The Proxy Chain Method

In three-dimensional space, the proxy can be locally constrained in three independent directions. As illustrated in Figure 6.6, depending on the number of constrained degrees of freedom, a point, curve, or surface constraint is obtained. As discussed in Section 6.3, when using the linear model, compound effects can be constructed by controlling the motion of the proxy along orthogonal reference directions. Nonlinear constraints, however, refine the location of the proxy iteratively. Thus, for nonlinear constraints it is not possible

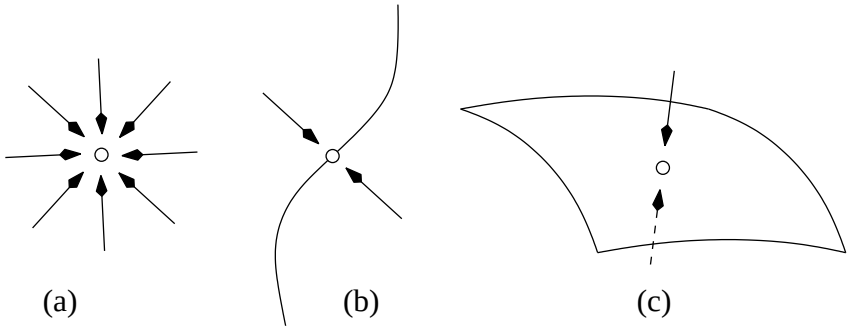


Figure 6.6. Basic types of three-dimensional constraints: (a) point, (b) curve, and (c) surface constraint.

to use the orthogonal frame approach for combining multiple intersecting or overlapping constraints.

To illustrate the problem, consider the combination of a penetrable isosurface constraint and a streamline curve constraint. In the example shown in Figure 6.7, the data probe is constrained to the curve until the proxy penetrates the surface. The goal is to follow to curve inside the surface and simultaneously provide haptic cues about the penetration depth. When the constraint pipe method is used [71], the proxy stays fixed at the intersection of the two constraints until the distance between the probe and the proxy reaches a user-defined threshold and the surface constraint is deactivated. As shown in Figure 6.7(a), both constraints remain satisfied, but no haptic cues are generated indicating the penetration depth into the surface. In addition, depending on the relative orientation of the two constraints, the surface constraint may be deactivated even if the probe only slightly penetrates the surface. With the force-based approach [113], shown in Figure 6.7(b), the location of the proxy is updated by balancing the forces from the probe, the surface constraint and the curve constraint. Since the update step uses only local information from the two fields and it attempts to globally balance the penalty forces obtained from violating the curve and the surface constraint, the proxy will drift from both constraints. In fact, there is no guarantee that the proxy will satisfy any of the constraints. Thus, it is not possible to continue exploring the original streamline curve after the surface constraint is deactivated, because the proxy will be located on a different streamline.

There is no general proxy update strategy that works for all applications [71]. One possible choice is to always select the constraint closest to the probe [83]. This approach

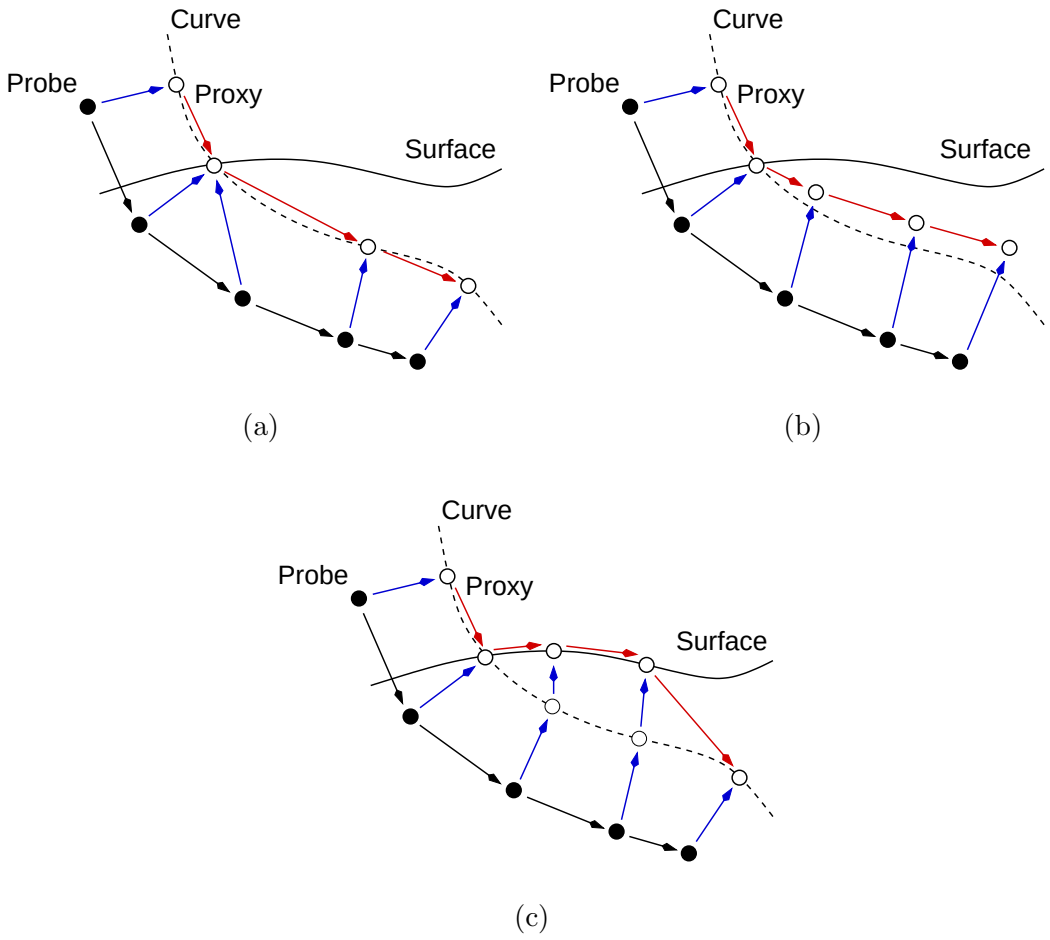


Figure 6.7. Motivation for the proxy chain method. In this example, a penetrable isosurface constraint is combined with a streamline curve constraint. The probe is constrained to the curve until penetrating the surface. (a) When the constraint pipe method is used, the proxy stays fixed at the intersection of the two constraints until the distance between the probe and the proxy reaches a user-defined threshold and the surface constraint is deactivated. Note that depending on the relative orientation of the two constraints, the surface constraint may be deactivated even if the probe only slightly penetrates the surface. (b) With the force-based approach, the proxy is updated by balancing the forces from the probe, the surface constraint and the curve constraint. Note that after penetrating the surface, the proxy will drift from both constraints. (c) By keeping a separate proxy for each constraint, the probe can be constrained to the curve and the surface simultaneously. The surface constraint is deactivated when the distance between the curve proxy and the surface proxy reaches a user defined limit, resulting in a better approximation to a penetrable surface of finite depth.

is useful for building simple compound constraints that feel continuous, such as a curve constraint from connected line segments or a surface constraint from triangle primitives. However, the approach suffers from the same problems as the force field method [190]. In addition, it is not possible to implement intersecting constraints without adding lower dimensional primitives to the scene.

The solution to the problem of combining a penetrable isosurface constraint and a streamline curve constraint is to maintain two separate proxies for the constraints. The proxies are updated in series during the proxy update step to obtain the desired haptic effect. Thus, instead of finding a single proxy location that attempts to satisfy all of the constraints simultaneously, the proxies form a chain, such that the proxy found in the current step becomes the probe for the next step in the chain. As shown in Figure 6.7(c) by keeping a separate proxy for each constraint, the probe can be constrained to the curve and also provide feedback about the penetration depth. Thus, the proxy update step is composed of two parts. First, the curve proxy is updated using the new probe position. Next, the surface proxy is updated using the new curve proxy as the probe. The surface constraint is deactivated when the distance between the curve proxy and the surface proxy reaches a user defined limit, resulting in a better approximation to a penetrable surface of finite depth.

6.8 Summary

This chapter presented a haptic rendering technique based on volumetric constraints. The technique provides a general framework for building haptic modes for data probing tasks and augmenting user interface elements with guiding feedback. The method represents each constraint by a set of orthogonal directions. By using an orthonormal frame representation, a variety of linear constraints can be expressed, including point, line, and planar constraints. Motion rules provide a simple framework for creating haptic effects, such as a penetrable shell or surface texture effect. Nonlinear constraints facilitate accurate guidance for data exploration. Note that the nonlinear formulation also includes an associated reference frame, but the frame may vary during the proxy update step. Finally, the proxy chain algorithm can be used to effectively combine several nonlinear constraints without introducing drift and ambiguity between the visual and haptic feedback modes.

CHAPTER 7

CONTRIBUTION OF HAPTIC GUIDANCE TO VECTOR FIELD EXPLORATION TASKS

Haptic feedback has been shown advantageous as an interaction modality for augmenting data visualization and manipulation techniques [84, 5, 105]. In addition, human factors studies have demonstrated that haptic feedback improves the speed and accuracy of spatial exploration and learning tasks [176, 137, 3, 45]. For visualization tasks, the advantage of properly combined visual and haptic guidance is that the reduced cognitive workload allows the user to concentrate on the relevant aspects of the task or the data. In practice, it is important to develop a suitable guiding strategy to avoid user frustration [131].

To quantify the contribution of haptic guidance in data exploration tasks, a controlled experiment was conducted. The purpose of the study was to evaluate user performance for two representative tasks with and without haptic guidance and to examine the difference in difficulty between exploring 2D and 3D data sets. The chosen task domain was vector field visualization, because exploration tasks in this domain are suitable for objective evaluation.

7.1 Experimental Procedure

The tasks were designed based on related experiments from the literature [137, 45, 102]. The chosen tasks were representative of typical user interactions and were simple enough for repeated execution. For each task, subjects were presented with images of synthetic vector fields. Each image consisted of a sparse set of streamlines that indicated the global behavior of the field. In addition to the global view, subjects were required to use the haptic probe to learn more about the local properties of the data.

A stereoscopic desktop workstation and a SensAble PHANToM 3.0L was used for the experiment. The PHANToM was mounted on a 75 cm high table in desktop configuration.

Images were displayed at 1152×864 resolution on a $20''$ monitor placed on the table 50 cm left of the base of the haptic device. Subjects were seated so that their eyes were located at approximately 60 cm from the monitor surface. Images for the 3D tasks were displayed in stereo using CrystalEyes LCD shutter glasses. To eliminate reflection from external light sources on the inside surface of the glasses, all lights in the room were blocked. A grey background was chosen for the images to minimize ghosting artifacts from the shutter glasses. In addition, the refresh rate of the monitor was set to 120 Hz to eliminate flicker. The stereo view allowed users to perceive objects within an approximately $25 \times 25 \times 25$ cm^3 volume centered on the monitor surface. The haptic device was positioned such that subjects could conveniently move the stylus within a $40 \times 40 \times 40$ cm^3 volume located on the right hand side of the monitor. The stylus was represented with a crosshair cursor within the visual workspace. A keyboard was also added to collect input from the users without disturbing the motion of the haptic probe.

The structure of the experiment is illustrated in Figure 7.1. Participants were first given general instructions on how to use the experimental setup. At the beginning of each task, further instructions specific to the task were provided. Next, a practice session consisting of 10 trials were conducted, followed by a recorded session of 20 trials. At the beginning of each trial, subjects were asked to move the cursor to a starting location indicated by a blue dot on the screen. The haptic device provided active guidance by snapping the cursor to the starting point. However, due to the limited stiffness of the device, subjects also had to make sure that they visually aligned the cursor with the blue dot. Once the cursor was placed in the correct starting position, subjects proceeded with the trial by pressing the space bar. Subjects also indicated completion of the task with the space bar. At the end of each trial, a text screen was shown indicating that the subject could begin the next trial after an optional short break.

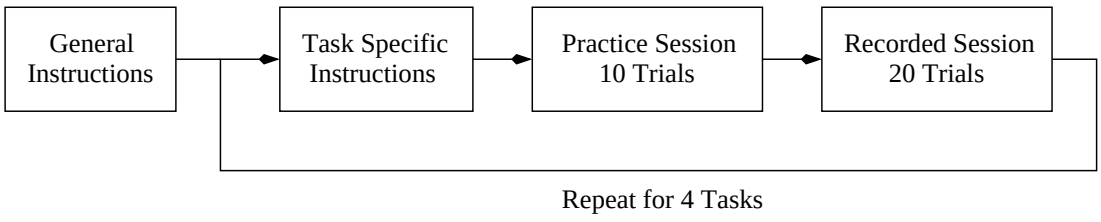


Figure 7.1. Organization of the experiment.

For the 2D advection task, subjects were asked to follow a local indicator, as shown in Figure 7.2(a). The starting position was selected so that the correct path would not match one of the global streamlines. The task was completed when the cursor hit the edge of the circle placed around the starting position, as shown in Figure 7.2(b). The radius of the circle was chosen such that the correct path had a constant length for every trial. A short streamline segment was advected from the cursor position with an arrowhead placed at the end, indicating the local direction of the flow. Users were instructed to follow the local indicator as accurately as possible in a single sweeping motion. In addition, they were instructed to complete the task as quickly as possible and were given a 10 second time limit.

For the 2D tracing task, subjects had to follow a static path with the cursor, as illustrated in Figure 7.2(c). The path was visually represented by a red streamline segment originating from the starting point. The task was completed when the cursor reached the target at end of the segment, as shown in Figure 7.2(d). Users were instructed to trace the path with the cursor as accurately as possible and to keep the cursor on the streamline all the time. Completion time was restricted to 10 seconds for this task as well.

The 3D tasks had a similar structure to the 2D tasks, as shown by Figure 7.3. The most important difference was that a plane target was used for the advection task rather than a hemisphere. A plane was chosen, because intersecting a semitransparent hemisphere was deemed too difficult during the pilot studies. The plane was always oriented as a ground plane and was textured with a checkerboard pattern. Several other measures were taken to improve depth perception of the scene: depth cuing using fog and lighting cues was added, and the starting position was chosen to minimize occlusion of the path by the global streamlines. In addition, the path and the viewing orientation were selected such that most of the cursor motion was parallel with the image plane of the monitor. Finally, a haptic cue was added to indicate the intersection of the cursor with the plane.

Two different haptic stimuli were used in the experiment. For the ISO condition, a slight isotropic drag force was added using a point constraint. The isotropic feedback was helpful, because it kept the probe in the same location when users released the stylus and stabilized the motion of the probe during the tasks. For the ANISO condition, probe motion was restricted to the correct streamline using a nonlinear constraint, as described in Chapter 6, Section 6.6. The anisotropic feedback helped users to keep the probe on the path. However, they still had to make sure the cursor visually intersected the streamline,

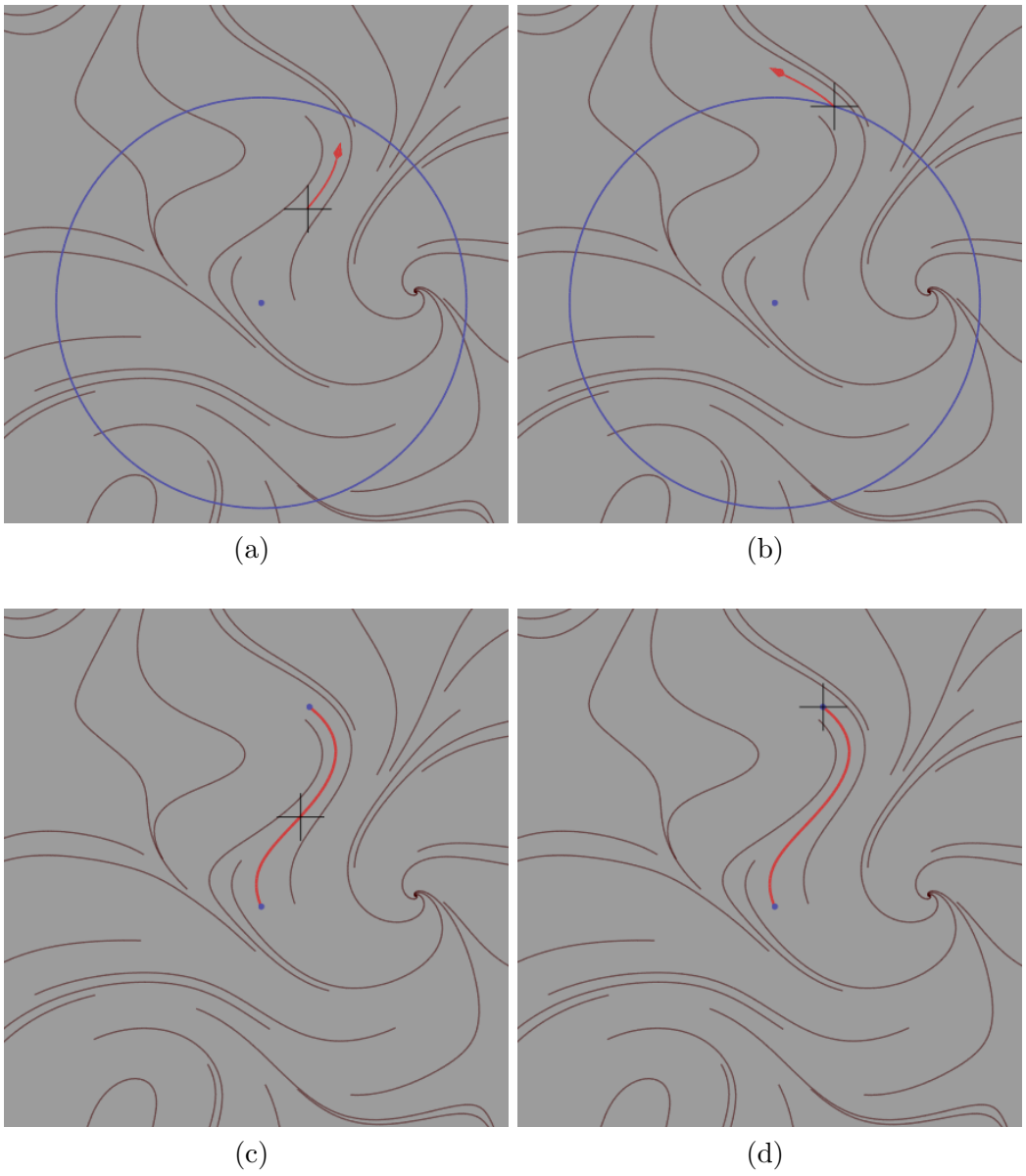
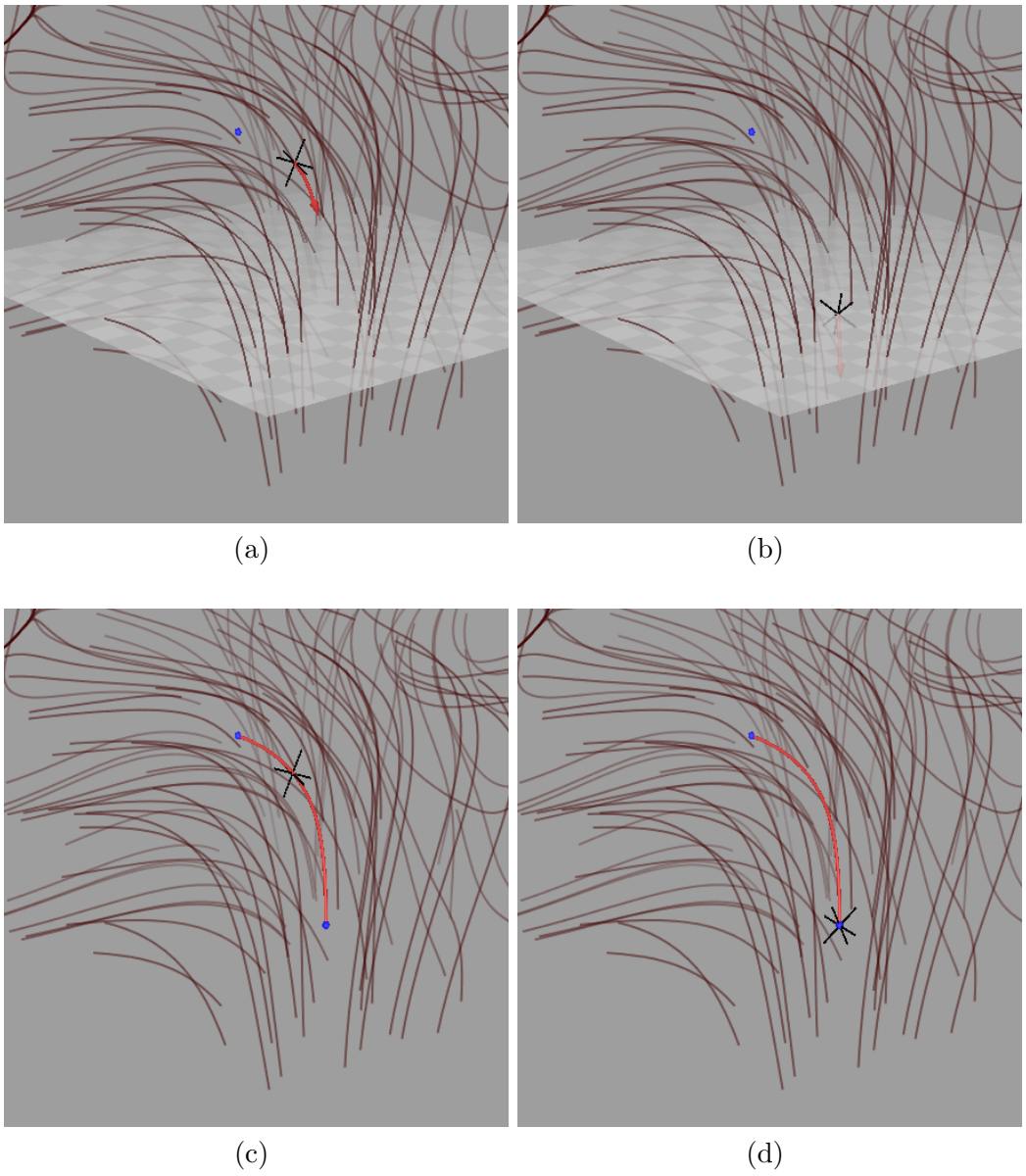


Figure 7.2. Illustration of the two-dimensional vector field exploration tasks. (a) During the advection task, subjects followed a local streamline indicator with the cursor from the starting position until (b) the cursor hit the edge of the circle placed around the initial position. (c) During the tracing task, subjects followed a static streamline segment with the cursor from the starting location until (d) the cursor reached the endpoint of the segment.



(a) (b)

(c) (d)

Figure 7.3. Illustration of the three-dimensional vector field exploration tasks. (a) During the advection task, subjects followed a local streamline indicator with the cursor from the starting location until (b) the cursor hit the ground plane placed below the starting position in the data. (c) During the tracing task, subjects followed a static streamline segment with the cursor from the starting location until (d) the cursor reached the endpoint of the segment.

because with some effort it was still possible to move the cursor off the path due to the limited stiffness of the haptic interface. Finally, to simplify the procedure, only a single haptic constraint was included in the tasks, with the exception of the 3D advection task. For this task, the proxy chain method was used for combining the anisotropic guiding constraint with a plane constraint to generate a haptic cue at the intersection of the selected streamline and the ground plane.

The 2D vector fields were generated using a technique similar to a previously published method [102]. Each 2D data set was created using the `griddata` function in Matlab with the biharmonic spline interpolation (`v4`) option from 19 randomly selected vectors on a 128×128 uniform grid. Preliminary evaluation of the fields indicated that 19 seed vectors were sufficient to obtain fields with a variety of features for both tasks. Since Matlab does not provide smooth scattered data interpolation functions for more than two dimensions, custom software was written for the 3D data sets. The scattered data interpolation function was based on the inverse distance of the gridpoint from the seed location. The data sets were generated on a $32 \times 32 \times 32$ grid from 13 randomly selected seed vectors. Using more seedpoints resulted in fields that were too cluttered for the 3D stereoscopic view. The size of the grid was selected to match the screen space complexity of the distribution of the data locations of the 2D data sets. A total number of 30 data sets were generated for both cases.

A sparse set of streamlines seeded on a uniform grid was selected for the global visualization technique. Seed points were placed on a 6×6 and $5 \times 5 \times 5$ grid, respectively. A third-order Runge-Kutta method with fourth-order adaptive stepsize control was used for particle advection [139]. Between 70 and 90 starting locations and viewing orientations were selected for each task, out of which only the 30 best were kept. By precomuting the streamlines and storing them in a display list, it was possible to maintain a sufficiently high 60 Hz visual update rate. No perceptible delay between the visual and haptic feedback was noticed during the implementation and the pilot studies. A randomized latin square design [119] was used to ensure that each subject was presented with an equal number of trials with the two haptic conditions, and that both the ANISO and ISO versions of the same trial were equally distributed among the users.

Fifteen male and five female participants were recruited from the Scientific Computing and Imaging Institute and School of Computing student and staff pool. The experiment did not require the subjects to know anything about flow visualization techniques. Most

participants, however, had some experience with vector fields from their classes. One of the subjects was an active researcher in the field. The age of the participants varied between 22 and 44 years and all of them were right handed.

Task completion time and position error were used as performance measures for the evaluation. For the advection tasks, position error was defined as the magnitude of the difference between the correct intersection point and the one found by the user. For the tracing tasks, position error was defined as the average deviation from the correct path. The deviation was computed as the distance between the cursor and the closest point on the curve.

Not all trials were considered in the analysis. Trials with unusually high completion times or position errors were discarded. These trials correspond to the cases when users got lost in the data. In total, 70 of the 1600 trials were rejected based on cut-off error limits (30 mm for 2D and 100 mm for 3D) and time limits (10 and 15 seconds for the advection tasks and 12 and 17 seconds for the tracing tasks). The time cutoffs for the tracing tasks were increased, because users spent more time completing these tasks. The means and standard deviations of the performance measures for each user were input for statistical analysis of the data.

In addition to the measurements, informal feedback was collected after the experiment in the form of a questionnaire. The questions included whether subjects found the 3D trials more difficult than the 2D trials and if they used any particular strategy for completing the tasks.

7.2 Results and Analysis

Three different methods were used for analyzing the data. Pairwise t-tests for the means and standard deviations of both measures between the ANISO and ISO conditions provided direct comparison of subject performance. The t-tests were augmented with graphical comparison of the condition means along with 95% confidence intervals. Based on the graphical comparison, two means can be considered statistically different, if the confidence intervals do not overlap. Finally, to examine the corellation between position error and completion time, per subject means were plotted for each task.

The results of the t-tests are summarized in Tables 7.1 and 7.2. One would think that the lack of constant global feedback made the advection task more difficult than the tracing task. However, the subjects spent more time on the tracing task, due to the direct visual feedback provided about their performance. Correspondingly, the subjects

Table 7.1. Comparative analysis of position errors. The results of the pairwise t-tests show that the differences between the means and standard deviations for the ANISO and ISO conditions are statistically significant.

Task	Mean	Std
2D Advection	$t(19) = 12.25, p < 0.001$	$t(19) = 8.75, p < 0.001$
2D Tracing	$t(19) = 11.14, p < 0.001$	$t(19) = 3.17, p < 0.005$
3D Advection	$t(19) = 8.12, p < 0.001$	$t(19) = 9.77, p < 0.001$
3D Tracing	$t(19) = 4.76, p < 0.001$	$t(19) = 3.1, p < 0.006$

Table 7.2. Comparative analysis of completion times. The results of the pairwise t-tests show that the differences between the means is statistically significant, which is not the case for the standard deviations.

Task	Mean	Std
2D Advection	$t(19) = 6.19, p < 0.001$	$t(19) = 2.65, p < 0.016$
2D Tracing	$t(19) = 6.93, p < 0.001$	$t(19) = 0.127, p < 0.9$
3D Advection	$t(19) = 3.76, p < 0.001$	$t(19) = 2.5, p < 0.022$
3D Tracing	$t(19) = 8.75, p < 0.001$	$t(19) = 1.58, p < 0.13$

felt that the tracing tasks were more difficult, which was indicated by the results of the postexperiment questionnaire. Since only limited performance feedback was provided during the advection tasks, making a conscious effort to find the optimal path was not possible. The experimental measurements reflect the observation of the users, because the tracing tasks took more time to complete with significantly better accuracy.

The graphs for comparing the condition means for the 2D and 3D tasks are shown in Figures 7.4 and 7.5, respectively. The results of the pairwise t-test analysis indicate that for all four tasks the ANISO condition improves both positioning accuracy and completion time significantly. The pooled analysis in Figures 7.4 and 7.5, however, shows no significant difference between task completion times, except for the 3D tracing task. The results indicate that the connection between the graphical and haptic feedback is easier to establish for the tracing tasks. In addition, many subjects reported that the difference between the ANISO and ISO conditions is more noticeable for the 3D tasks than the 2D tasks. Thus, the two conditions were easiest to distinguish from each other during the 3D tracing task. However, since there was no strict control established for the

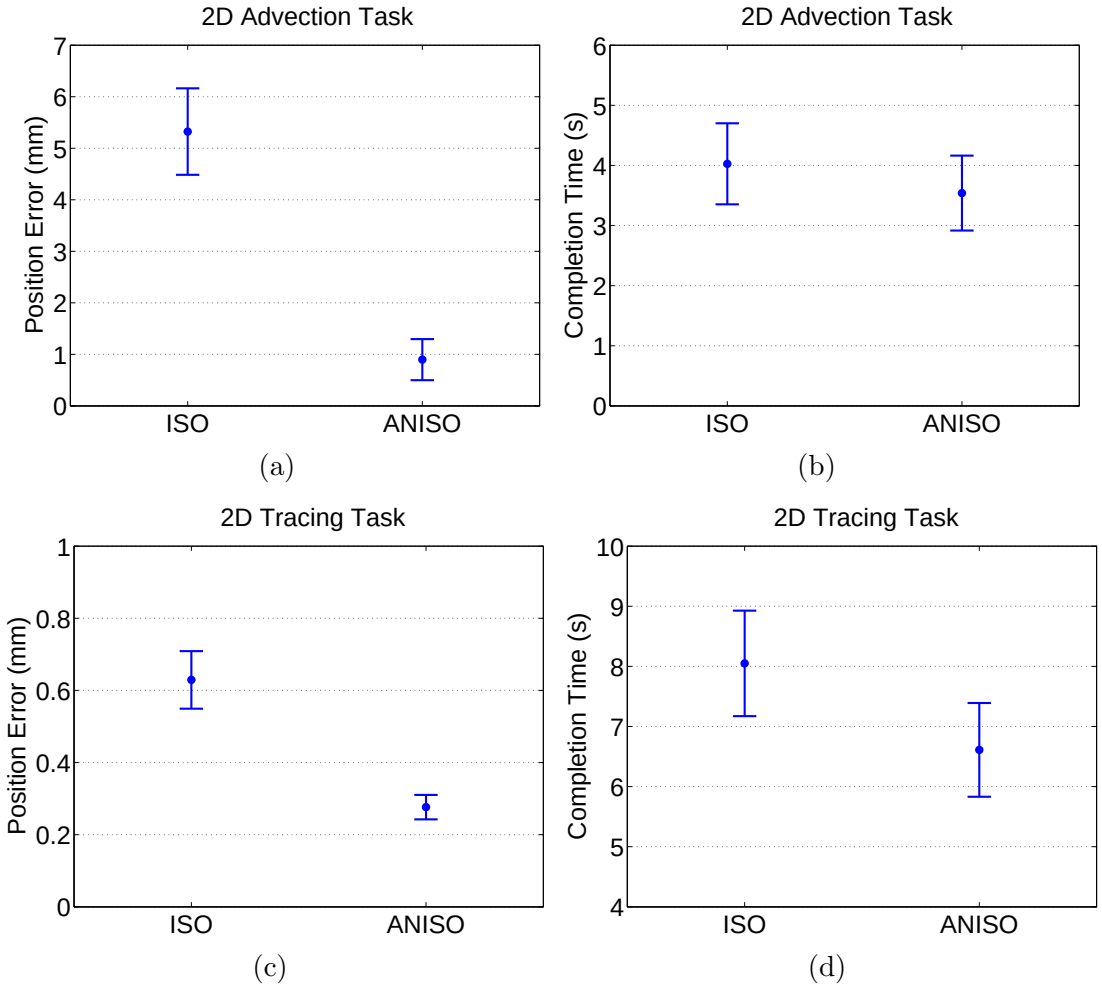


Figure 7.4. Mean position errors and completion times for the 2D tasks. The error bars show 95% inferential confidence intervals. Nonoverlapping bars indicate that there is a statistically significant difference between the means. The graphs show that both measures are lower for the ANISO condition than for the ISO condition. However, there is no significant difference between task completion times.

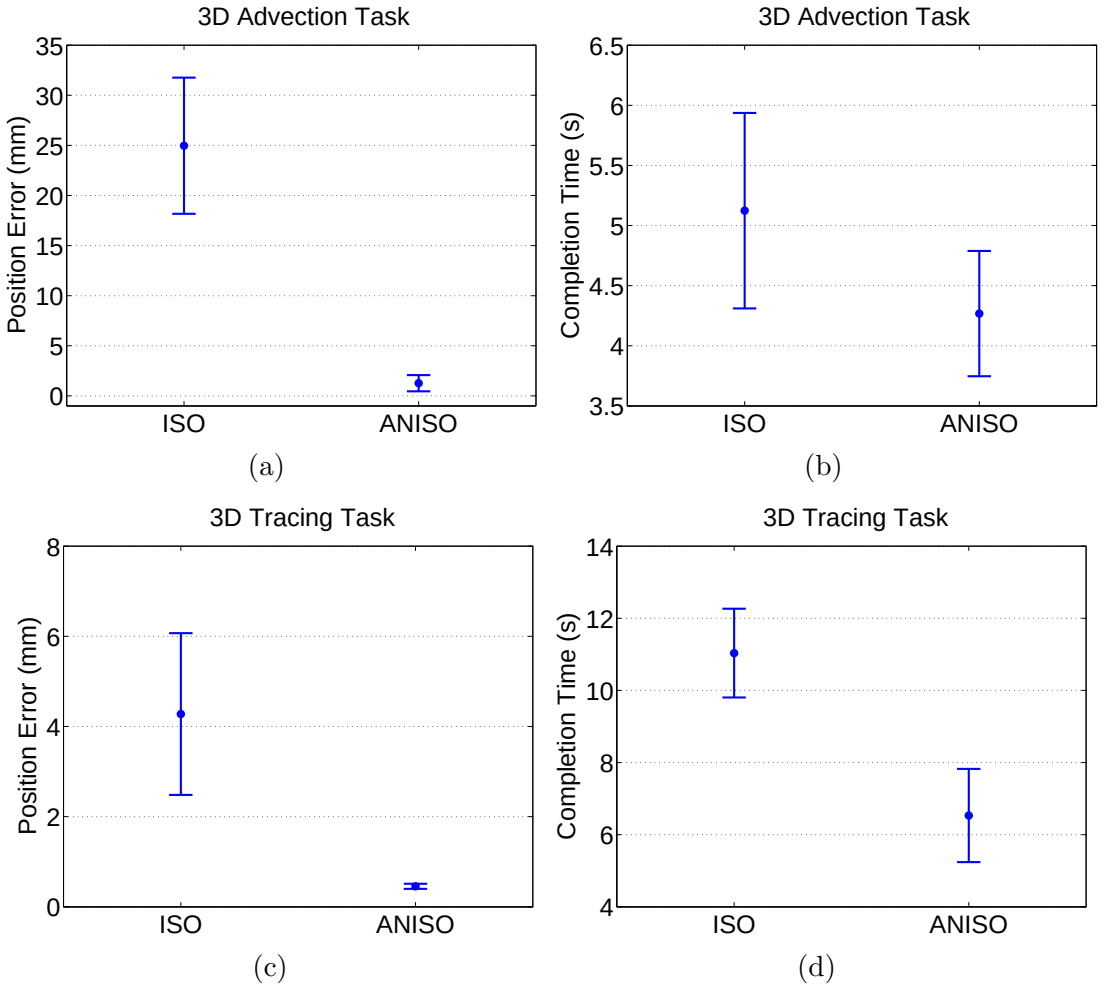


Figure 7.5. Mean position errors and completion times for the 3D tasks. The error bars show 95% inferential confidence intervals. Nonoverlapping bars indicate that there is a statistically significant difference between the means. The graphs show that both measures are lower for the ANISO condition than for the ISO condition. Completion time is significantly lower for the ANISO condition for the tracing task, but not for the advection task.

timing of the tasks and the advection task had limited feedback on task performance, general conclusions should not be drawn from the timing data, other than comparing the difficulty of the 2D and 3D tasks.

All users indicated in the postexperiment survey that the 3D tasks were significantly more difficult than the 2D tasks. Pairwise t-tests between the accuracy of the 2D and 3D tracing tasks confirm this claim, even with haptic feedback ($t(19)=7.38$, $p < 0.001$). Most participants complained about the effort needed to find the intersection of the 3D cursor with the streamline, due to the limited depth information provided by the graphical feedback. For intersection tasks, better local feedback could be provided by orthogonal transparent planes. However, finding the right balance between occlusion and transparency is not straightforward and can also be subjective.

For a few subjects depth perception was problematic. One subject had difficulty seeing the difference in depth between the starting and target locations on the streamline. Two other users experienced eye strain from focused viewing of the stereo imagery.

An important issue was speed of execution. Many users felt that with smoother and faster motions they performed better, even without haptic feedback. One subject, however, noticed a conditioning effect when several trials with the ANISO condition were present in a row.

In general, users appreciated the haptic guidance for both tasks. Several subjects reported that they used the strategy of slowly determining at the beginning of the trial if the guiding forces were present or not and using them to quickly complete the task. In contrast, others found that they relied completely on visual feedback, but felt that some trials were easier than others. Most users followed the local indicator to accomplish the advection tasks. However, they considered it only a rough guide. Only one user reported on the usefulness of global feedback for the 2D advection task. However, for the 3D case, the global feedback was found inefficient by the majority of the subjects.

Another important question is whether there was correlation between the timing and accuracy measures. To examine the corellation, per subject means were plotted for each task, as shown in Figure 7.6. The most significant correlation was established for the 2D tracing task with the ISO condition (Pearson: $-0.704, p < 0.001$). The general trend in the distributions corresponding to the ISO condition is that whenever users tried to accomplish the task faster, accuracy was lower. The trend was less significant for the ANISO condition.

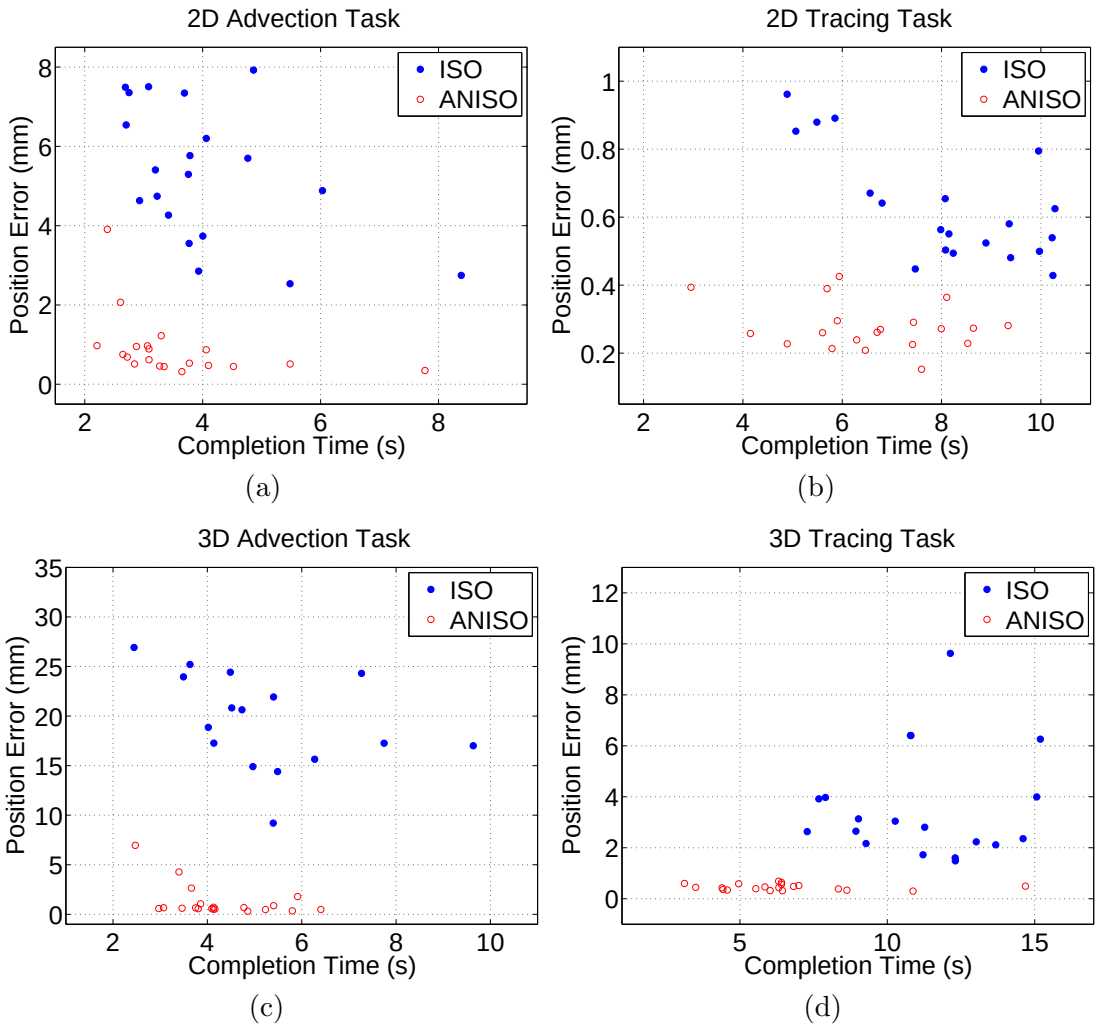


Figure 7.6. Distribution of position error versus completion time for the tasks. The results show that shorter completion times correspond to larger errors for the ISO condition, but not necessarily for the ANISO condition.

The outcome of the experiment confirms that the human visual and fine motor skills excel in two dimensions, especially when proper visual feedback is available. The importance of constrained feedback in three-dimensional precision tasks is reflected by the large differences between the ANISO and ISO conditions with respect to both positioning accuracy and timing of the tasks. When visual feedback on user performance was available, subjects made efforts to complete the tasks more accurately. Best results, however, were obtained when both constant visual and haptic feedback were provided to the users.

CHAPTER 8

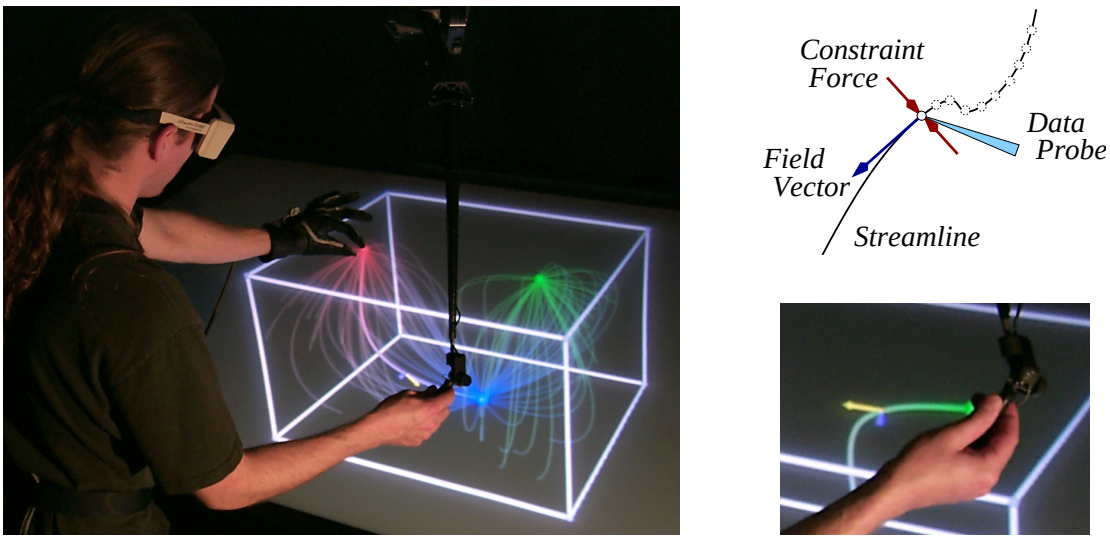
EXAMPLES

This chapter presents examples of combined visual and haptic exploration modes for scalar, vector, and tensor data. The goal of the combined feedback is to provide information about the local properties of the data in the form of reinforcing visual and haptic cues. In addition, some of the examples show how haptic constraints can help the user place and manipulate 3D user interface elements in the volume.

8.1 Comparing Exploration Modes for Vector Field Data

Streamline advection is a basic building block of vector and tensor field visualization techniques. Advection is based on integrating the motion of massless particles according to the velocities defined by the field. The haptic equivalent of a streamline is achieved by constraining proxy motion along the path of a single particle, as described in Section 6.6. Thus, any effort to move the probe in a direction perpendicular to the current orientation of the field results in a strong opposing force, as illustrated in Figure 8.1. If the user pushed the probe hard enough, the proxy could "pop over" to an adjacent streamline, allowing the user to move the probe in three dimensions and still receive strong haptic cues about the orientation of the flow. An additional force component can be used along the streamline to indicate the magnitude of the field. Alternatively, a secondary constraint can be added to convey information about the speed of the flow in the form of haptic tickmarks.

The constraint-based vector field exploration mode was evaluated during a research demonstration at the IEEE Visualization Conference. Visitors were asked to explore a vector field data set on the Visual Haptic Workbench, as shown in Figure 8.1(a). The demonstration application implemented three different exploration modes along with global and local vector field visualization methods. Illuminated streamlines were used as a global visualization technique to show the global nature of the field without visual



(a) (b)

Figure 8.1. Exploration of vector field data. (a) A user explores a volumetric vector field on the Visual Haptic Workbench. (b) The data probe is constrained along a streamline resulting in intuitive haptic feedback.

clutter [191]. The streamline under exploration was highlighted via a set of stream balls. A local vector icon indicated the direction and magnitude of the field at the haptic probe. The outcome of the demonstration indicated that the majority of the users preferred the constraint-based approach over the two force field rendering modes: the direct display mode [40] and the transverse damping mode [136].

8.2 Exploration of Cardiac Simulation Data

The haptic streamline constraint can be easily modified to display orientation information on isosurfaces. Such a technique could be useful for investigating the relationship between heart muscle fiber orientations and potential distributions resulting from bioelectric finite element simulations [127]. The simulations are typically carried out on a curvilinear grid created from a number of epicardial and endocardial heart muscle layers.

The implementation first transforms the curvilinear grid to a tetrahedral mesh by computing a Delaunay triangulation of the original data points. Next, scalar values are assigned to the nodes in individual layers in increasing order from inside to outside. Isosurfaces of the scalar field correspond to muscle layers in the model. Next, the gradient field of the scalar field is computed using a central difference approximation formula. The gradient field is needed for the iterative refinement to make sure the proxy

stays on the currently selected layer, as described in Section 6.6. To avoid singularities when interpolating fiber orientation vectors within a tetrahedral element, the method uses component-wise linear interpolation of the tensor field obtained by taking the outer product of the vectors with themselves. The major eigenvector of the interpolated tensor yields a smooth orientation field within a tetrahedral element, even when the vectors at the nodes point in completely different directions.

In this example, a local reference frame is formed by the interpolated fiber orientation and gradient vectors. Two proxies are linked in a chain to achieve the desired haptic effect. The first proxy $\mathbf{p}_s$ restricts probe motion to a selected isosurface of the scalar field. The second proxy $\mathbf{p}_c$ provides directional cues by adding a friction effect perpendicular to the tangential component of the direction field. The snap-drag motion rule provides the user with a means to explore a single layer, and "pop through" to a neighboring layer by pushing against the surface. Instead of using the drag threshold τ_i to move the proxy after it is released from the current layer, the implementation detects when the probe crosses a neighboring layer and set the proxy location to the intersection point. A secondary snap-drag rule constrains proxy motion along the fibers on the surface, allowing the user to switch to a nearby streamline in discrete steps. The method essentially creates a haptic texture on the surface composed of tiny valleys and ridges corresponding to the muscle fibers. See Figure 8.2 for an illustration of the technique.

Informal evaluation of the method concluded that haptic guidance is of little value, unless it precisely matches the visual feedback. Thus, it is important to provide visual indicators that support the haptic exploration mode by conveying the same information about the local properties of the data. For this example, accurate visualization of the direction field on the isosurface is important to avoid user confusion.

8.3 Anisotropic Friction Mode for Mixed Scalar and Vector Data

The method presented in the previous section can be augmented with a third component to provide not only visual, but also haptic cues about the local orientation of the direction field relative to a isosurface of the geometry scalar field. As shown in Figure 8.3(a), the anisotropic friction mode incorporates a third proxy in the chain. The first proxy $\mathbf{p}_s$ restricts probe motion to a selected isosurface of the scalar field. The second proxy $\mathbf{p}_c$ provides directional cues by adding a friction effect perpendicular to the

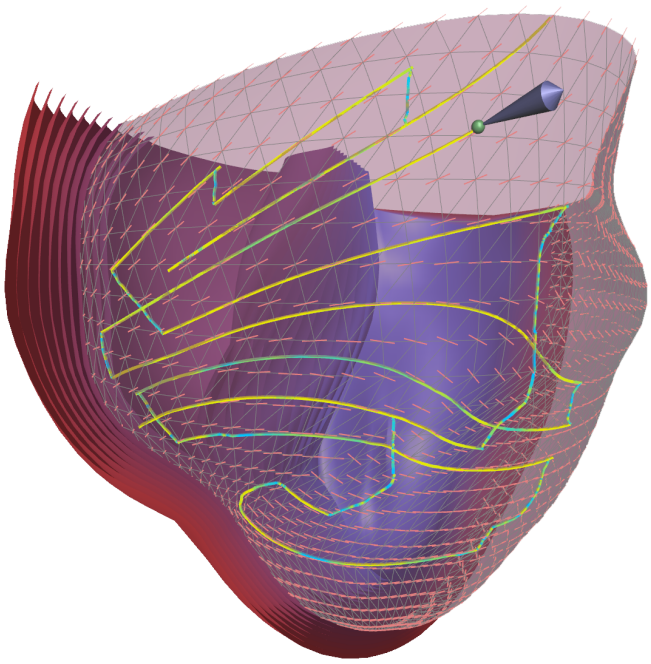


Figure 8.2. Exploring epicardial muscle fibers with haptic feedback. The probe is constrained to follow local fiber orientation on the surface of a single layer. The user can "pop through" to a neighboring layer by pushing against the surface. Similarly, the user can choose a different fiber by pushing perpendicular to the current one while staying on the surface. The effect feels as if the surface was textured with tiny valleys and ridges. The image shows the path of the proxy colored according to the magnitude of the applied force component perpendicular to the fiber orientation and tangent to the surface, from yellow to cyan indicating increasing tension between the probe and the proxy.

tangential component of the direction field. The third proxy $\mathbf{p}_f$ implements an additional frictional term that is modulated according to the orientation of the direction field relative to the surface. The simulated effect is that the more perpendicular the field vector $\mathbf{v}$ to the surface, the more difficult it is to drag the probe along the streamline. Hence, the drag threshold for the third proxy τ_f is a function of the field vector $\mathbf{v}$ and the surface gradient ∇s at $\mathbf{p}_f$,

$$\tau_f = u\,\tau_{min} + (1-u)\,\tau_{max} \tag{8.1}$$

$$u = \cos\theta = [\ \nabla s \cdot \mathbf{v}\ ]\ \sigma(\|\nabla s\|\|\mathbf{v}\|) \tag{8.2}$$

where τ_{min} and τ_{max} are parameters for the minimum and maximum frictional drag threshold, respectively, and $\sigma(x)$ is a safe normalization term that ensures that the interpolation factor u reduces to zero in ill-defined regions of the two fields.

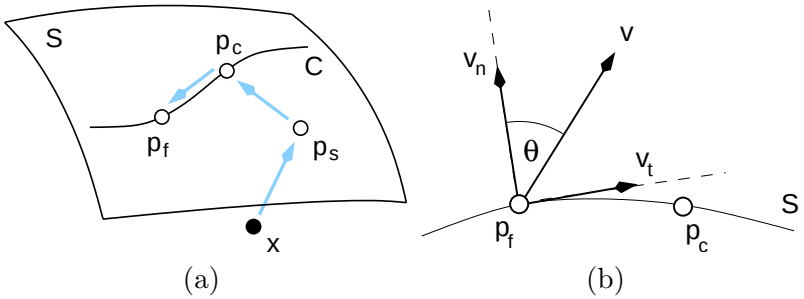


Figure 8.3. Anisotropic friction mode for exploring mixed scalar and vector fields. (a) By linking three proxies in a chain, the probe can be constrained to an isosurface of the scalar field and a streamline defined by the tangential component of the vector field. (b) The third proxy implements a friction effect that is modulated according to the orientation of the vector field relative to the surface. The field vector is decomposed into tangential and normal components. The resulting haptic effect is that the more perpendicular the field vector is to the surface, the more difficult it is to follow the streamline with the probe. Note that all proxies are located on the surface.

8.4 Exploration of Diffusion Tensor Fields

Diffusion tensor fields are difficult to visualize due to the increased dimensionality of the data values and complexity of the features that the values represent. Direct methods, such as glyphs and reaction-diffusion textures work well on two-dimensional slices, but are less successful for three-dimensional data sets. Intermediate representations created by adaptations of vector field techniques result in intuitive visual representations, but fail to capture every aspect of the field. Interactive exploration has been reported to be an effective means for helping users interpret the complex geometric models that capture important features in the data [188]. The exploration process can be further enhanced by adding haptic cues that guide the user according to the local orientation and anisotropy of the tensor field.

The rate and directionality of water diffusion in tissues is indicated by a second-order symmetric tensor. Anisotropy of the diffusion process can be characterized by the following barycentric measures [90]:

$$c_l = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2 + \lambda_3} \tag{8.3}$$

$$c_p = \frac{2(\lambda_2 - \lambda_3)}{\lambda_1 + \lambda_2 + \lambda_3} \tag{8.4}$$

$$c_s = \frac{3\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3} = 1 - c_l - c_p \tag{8.5}$$

where $\lambda_1 \geq \lambda_2 \geq \lambda_3$ are the sorted eigenvalues of the diffusion tensor matrix. The anisotropy measures indicate the degree of linear, planar, and spherical anisotropy, respec-

tively. The associated eigenvectors $\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{e}_3$ form an orthonormal frame corresponding to the direction of diffusion. Regions with linear and planar anisotropy represent important features in the data, such as white matter fiber bundles in brain tissue.

The constraint-based technique described in Chapter 6 can be adapted to indicate tensor orientation and degree of anisotropy through haptic feedback. The goal is to specify the motion rules such that the proxy is allowed to move along the major eigenvector, while being constrained in the other two eigendirections according to the degree of linear and planar anisotropy. For example, setting the drag thresholds to the following functions of the the anisotropy measures results in the desired feedback:

$$\tau_1 = 0 \tag{8.6}$$

$$\tau_2 = \tau(c_l) \tag{8.7}$$

$$\tau_3 = \tau(c_l + c_p) \tag{8.8}$$

where $\tau(x)$ is a monotonically increasing function on $[0 \ldots \tau_{\max}]$. The reason for the above choice is that the transfer functions result a line constraint along the major eigenvector in regions with linear anisotropy ($c_l \gg c_p, c_s$), a plane constraint in regions with planar anisotropy ($c_p \gg c_l, c_s$), and allow free motion along all three directions in isotropic areas ($c_s \gg c_p, c_l$). Note that the three measures sum to one, so when any one measure dominates, the transfer functions will emphasize the corresponding type of anisotropy. An alternative construction for the drag thresholds restricts motion of the proxy to a point in isotropic areas:

$$\tau_1 = \tau(1 - c_l - c_p) \tag{8.9}$$

$$\tau_2 = \tau(1 - c_p) \tag{8.10}$$

$$\tau_3 = \tau(1) \tag{8.11}$$

Note that the problem of undefined reference vectors in isotropic areas is resolved by the above definitions. A linear ramp suffices for $\tau(x)$, but other possibilities may also be appropriate.

The technique is illustrated in Figure 8.4. Users of the system found that it takes little effort to trace curves indicating fiber distribution and connectivity. In contrast, numerical methods for fiber tractography require careful selection of initial and stopping conditions and are not straightforward to use for investigating connectivity of regions in the data.

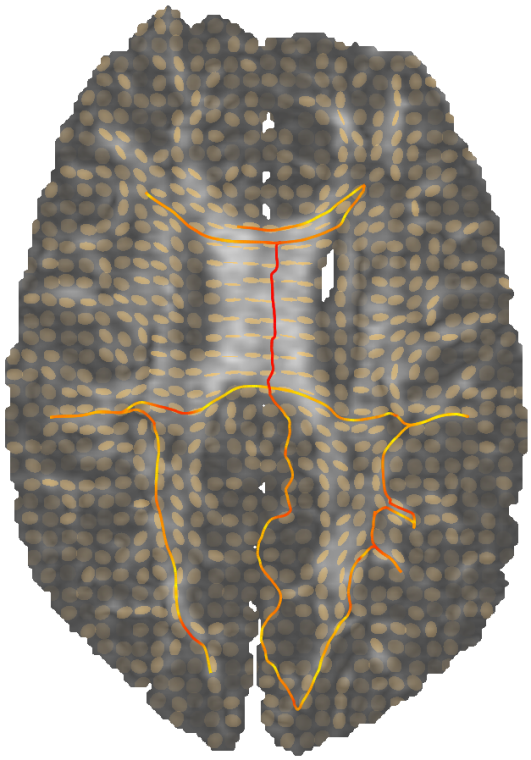


Figure 8.4. Exploring a DT-MRI data set with haptic feedback. The ellipses represent local diffusion anisotropy and orientation. Lighter areas have higher associated anisotropy. The proxy path is colored according to the magnitude of the applied force, from yellow to red indicating a larger tension between the probe and the proxy. The curves are tangent to the direction of the major eigenvector of the diffusion tensor matrix in anisotropic areas.

8.5 Data Exploration on a Haptic Grid

Perhaps one of the most straightforward uses of haptic constraints is to restrict probe motion along predefined directions, such as along the axes of a uniform grid, as shown in Figure 8.5. The method is the 3D equivalent of the snap to grid functionality found in popular computer-aided drawing and design packages. The possible uses of a haptic grid include placement of clipping planes, selection of a region of interest, dual-domain probing, and camera control. While similar functionality can also be provided through mouse input, it requires frequent rotations of the volume to accomodate all possible combinations.

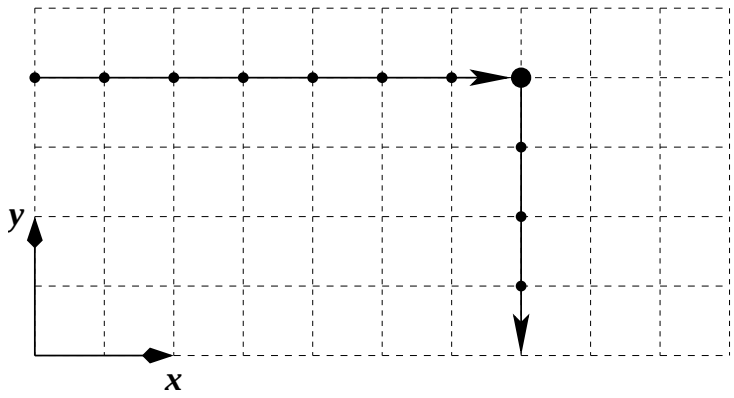
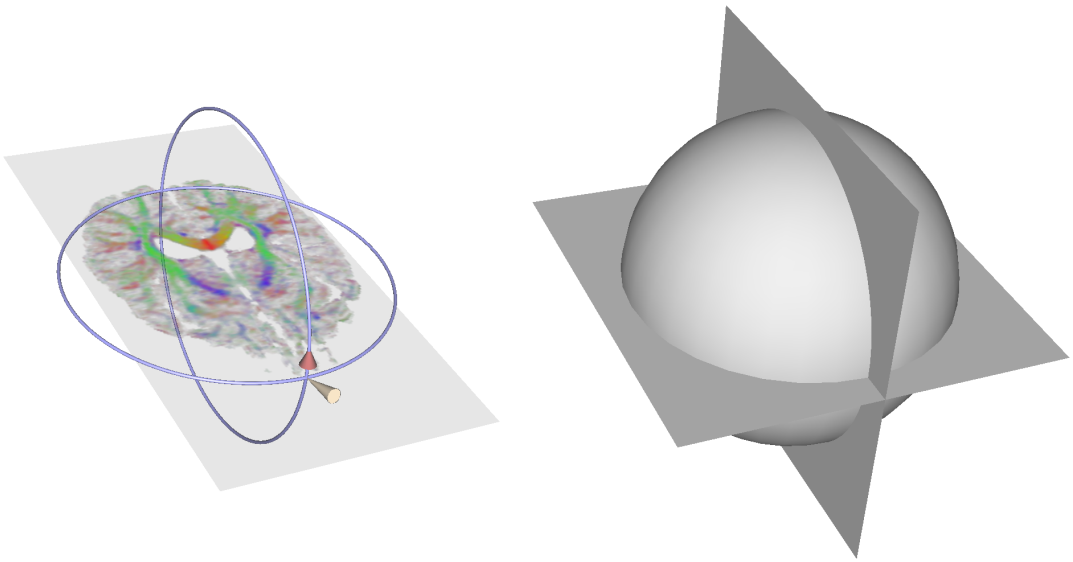


Figure 8.5. Example of data exploration on a regular grid. Probe motion is restricted along a selected axis until the user switches to a different axis by pulling the probe along the selected direction. Haptic landmarks can be placed along each axis to indicate possible transition points.

8.6 Constrained Haptic Modes for 3D User Interfaces

User interface elements can greatly benefit from constrained haptic modes [95]. For example, a 3D widget can provide intuitive and precise control over the position and orientation of cutting planes in the volume. The widget includes a spherical constraint and three planar constraints, as shown in Figure 8.6. The active constraints are ordered in the proxy chain according to the current mode of operation. For example, initially the user can snap the probe to the cutting plane to explore the data values on the surface of the plane. The user activates translation mode by pressing a button. In the translation mode, the cutting plane can be repositioned along three orthogonal directions and their combinations. A constraint plane is activated based on the distance of the probe from the corresponding proxy location. Note that the order of the plane constraints in the chain is not important, because the planes are orthogonal to each other.

When the probe is in proximity to the sphere, rotational mode is activated. In rotational mode, the sphere constraint is applied last in the proxy chain after the planar constraints. The advantage of using the proxy chain is that both axis-constrained and unconstrained rotational modes can be implemented without having to add embedded lower dimensional circle, line, and point constraints to the widget. In addition, it is easy to switch between modes of operation by activating, deactivating, and reordering constraints in the chain.



(a) (b)

Figure 8.6. Example of a user interface element with haptic feedback. (a) A haptic widget can be used for specifying the position and orientation of cutting planes in a volume. (b) The corresponding active constraints in the widget. In the configuration shown above, only two out of the three planes are active. The widget provides support for both axis-constrained and unconstrained translations and rotations.

CHAPTER 9

CONCLUSIONS AND FUTURE WORK

This dissertation has developed and evaluated global and local visualization techniques for the exploration of volumetric data sets. The methods were implemented for a testbed three-dimensional haptic virtual environment, the Visual Haptic Workbench [20, 75]. To provide maximum flexibility to the user during exploration, the presented visual and haptic data rendering techniques do not require a preprocessing stage to extract intermediate geometric representations from the data. Thus, the user has direct control over the visual and haptic display via a set of visualization parameters. In addition, exploration is supported at arbitrary scales, because the techniques are not restricted by the resolution of the intermediate geometric models.

Providing users with comfortable and effective immersive data exploration tools requires accurate tracking and display devices [171]. Chapter 4 presented calibration and registration techniques for improving and quantifying the display accuracy of the Visual Haptic Workbench. The methods reduce the overall static geometric error in the system by calibrating and colocating the tracking and display components. The coregistration procedure is fast, robust and flexible, and can be easily adapted to other configurations. It would be particularly interesting to evaluate the accuracy of a desktop system, such as the commercially available display from Reachin Technologies [1].

The provided analysis of head-tracking errors indicates that position errors cause the perceived virtual space to appear sheared and scaled. In addition, the analysis found that head orientation errors do not contribute significantly to visual distortion near the projection surface in front of the user. Unfortunately, correcting for static errors only partially solves the registration problem. It has been shown that end-to-end latency can be a significant contributing source of registration errors [69]. Future work might examine how dynamic registration accuracy can be improved via predictive filtering and synchronized display methods. Additional future improvements for the tracker calibration technique include the development of analytical error models obtained directly from the

operating principles of magnetic tracking devices and the combination of local and global error correction methods.

Chapter 5 presented three techniques for improving the interactivity of hardware-accelerated volume rendering for data exploration tasks. The methods attempt to reduce the number of fragment operations required to redraw the volume. A particular strength of the presented techniques is that the attainable speedup does not depend on the transfer function or the data. In addition, the focus and context decomposition method facilitates the development of novel visualization approaches that provide additional insight during exploration. Decomposing the workspace into focus and context regions is particularly suitable for immersive environment displays.

Future research on the focus and context decomposition technique might examine how the transition region could be used for creating more effective illustrations. One example in this direction is the ClearView system [98]. ClearView uses view and data dependent importance measures to fade out structures in the context in proximity of the focus. The transition region could also be useful for interpolating between two different resolution levels for time-critical display of isosurfaces. Combining the method with early ray termination and empty space skipping acceleration would further improve rendering performance. However, the transfer functions need to be chosen carefully for the region in which the acceleration techniques are used. Finally, the combination of the three presented methods would require a flexible and efficient GPU memory management scheme, such as the one offered by the Glift system [106].

Conducting a thorough analysis of the tradeoffs of the parameters used in hardware-accelerated volume rendering would be a valuable research contribution. There are many factors this dissertation did not take into account that could be used to further lower the number of fragments required for drawing the volume. For example, the sampling rate could be reduced for regions farther away from the viewpoint. Even though the performance of graphics accelerators will continue to increase rapidly in the future, so will dataset sizes and screen resolutions. By quantifying how the various components affect quality and performance, it will be possible to specify rendering parameters in a semi-automatic fashion.

A general constraint-based haptic rendering algorithm was described in Chapter 6. The algorithm is suitable for augmenting data exploration tools with haptic feedback, as illustrated by the examples in Chapter 8. The constraint-based haptic data rendering

approach has several desired properties: it provides a unified framework for different data modalities, secondary effects such as texture and friction are easily realized, haptic transfer functions are intrinsic to the algorithm, and control parameters can be tuned to the operational characteristics of the interface device. In addition, the algorithm can properly handle intersecting or overlapping nonlinear constraints, in contrast to previous methods [113].

A particular challenge for future research is the problem of synthesizing useful haptic transfer functions from the data. Creating haptic representations for noisy data is also an important research topic. A promising approach is to display the noise in the data in the form of secondary haptic effects, such as surface texture. A major limitation of existing constraint-based data rendering techniques is that the coupling between the visual and haptic feedback is restricted to a single point in the data. In contrast, force field methods have been extended to 6DOF haptic devices and various probe shapes [116]. However, it is not clear whether constraining torques would provide any benefit in data exploration tasks. Further extensions include the development of haptic rendering algorithms for multiresolution and point data sets and the design and evaluation of additional haptic user interfaces elements for immersive visualization applications.

Transforming the constraint-based approach to a purely functional formulation would provide a very natural space for specifying haptic rendering parameters. A disadvantage of using the spring-damper form of virtual coupling is that it is too conservative, which may limit the efficient display of secondary haptic effects. A promising alternative is an energy-based formulation that uses a time-domain passivity observer and controller to adaptively adjust the coupling parameters [60].

The experiment presented in Chapter 7 is an initial step towards quantifying the contribution of haptic guidance in data exploration tasks. The outcome of the experiment indicates that constraint-based haptic feedback improves positioning accuracy for representative vector field exploration tasks. In addition, the experiment confirmes the hypothesis that multimodal perception is optimal when both constant visual and haptic feedback are provided to the users. In the future, similar studies are needed to examine and quantify the benefit of adding haptic feedback to data exploration and manipulation tasks, especially relative to traditional desktop and immersive user interface implementations.

REFERENCES

[1] Reachin Technologies AB. `http://www.reachin.se/`.

[2] Adams, R. J., and Hannaford, B. Stable haptic interaction with virtual environments. *IEEE Transactions on Robotics and Automation 15*, 3 (1999), 465–474.

[3] Adams, R. J., Klowden, D., and Hannaford, B. Virtual training for a manual assembly task. *The Electronic Journal of Haptics Research 2*, 2 (2001).

[4] Arsenault, R., and Ware, C. Eye-hand co-ordination with force feedback. In *Proceedings ACM SIGCHI Conference* (The Hague, The Netherlands, Apr. 2000), pp. 408–414.

[5] Avila, R. S., and Sobierajski, L. M. A haptic interaction method for volume visualization. In *Proceedings IEEE Visualization Conference* (San Francisco, California, Oct. 1996), pp. 197–204.

[6] Aviles, W. A., and Ranta, J. F. Haptic interaction with geoscientific data. In *Proceedings PHANToM Users Group Workshop* (Aspen, Colorado, Oct. 1999).

[7] Azuma, R., and Bishop, G. Improving static and dynamic registration in an optical see-through hmd. In *Proceedings ACM SIGGRAPH Conference* (Orlando, Florida, July 1994), pp. 197–204.

[8] Bartz, D., and Gürvit, Ö. Haptic navigation in volumetric datasets. In *Proceedings PHANToM Users Research Symposium* (Zurich, Switzerland, July 2000).

[9] Bhaniramka, P., and Demange, Y. OpenGL Volumizer: A toolkit for high quality volume rendering of large data sets. In *Proceedings IEEE Volume Visualization and Graphics Symposium* (Boston, Massachusetts, Oct. 2002), pp. 45–53.

[10] Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T. D. Toolglass and magic lenses: The see-through interface. In *Proceedings ACM SIGGRAPH Conference* (Anaheim, California, Aug. 1993), pp. 73–80.

[11] Bierbaum, A., Just, C., Hartling, P., Meinert, K., Baker, A., and Cruz-Neira, C. VR Juggler: A virtual platform for virtual reality application development. In *Proceedings IEEE Virtual Reality Conference* (Yokohama, Japan, Mar. 2001), pp. 89–96.

[12] Blezek, D. J., and Robb, R. A. Haptic rendering of isosurfaces directly from medical images. In *Proceedings Medicine Meets Virtual Reality Conference* (San Francisco, California, Jan. 1999), pp. 67–73.

[13] Bouguila, L., Ishii, M., and Sato, M. Effect of coupling haptics and stereopsis on depth perception in virtual environment. In *Proceedings Workshop on Haptic Human Computer Interaction* (Glasgow, Scotland, Sept. 2000), pp. 54–62.

[14] Bowman, D. A., Kruijff, E., LaViola, Jr., J. J., and Poupyrev, I. An introduction to 3D user interface design. *Presence: Teleoperators and Virtual Environments 10*, 1 (2001), 96–108.

[15] Bowman, D. A., Rhoton, C. J., and Pinho, M. S. Text input techniques for immersive virtual environments: an empirical comparison. In *Proceedings Human Factors and Ergonomics Society Annual Meeting* (2002), pp. 2154–2158.

[16] Bowman, D. A., and Wingrave, C. A. Design and evaluation of menu systems for immersive virtual environments. In *Proceedings IEEE Virtual Reality Conference* (Yokohama, Japan, Mar. 2001), pp. 149–156.

[17] Boyles, M., and Fang, S. 3DIVE: An immersive environment for interactive volume data exploration. In *Proceedings International Conference on Computer Aided Design and Computer Graphics* (Yunnan, China, Aug. 2001), pp. 573–580.

[18] Brady, R., Pixton, J., Baxter, G., Moran, P., Potter, C. S., Carragher, B., and Belmont, A. Crumbs: A virtual environment tracking tool for biological imaging. In *Proceedings IEEE Biomedical Visualization* (Atlanta, GA, 1995), pp. 18–26.

[19] Brederson, J. D. The I^3Stick: An inexpensive, immersive, interaction device. Technical Report UUCS-99-016, School of Computing, University of Utah, Nov. 1999.

[20] Brederson, J. D., Ikits, M., Johnson, C. R., and Hansen, C. D. The Visual Haptic Workbench. In *Proceedings PHANToM Users Group Workshop* (Aspen, Colorado, Oct. 2000), pp. 46–49.

[21] Bresnahan, J., Insley, J., and Papka, M. E. Interacting with scientific visualizations: User-interface tools within spatially immersive displays. Technical Report ANL/MCS-P789-0100, Argonne National Laboratory, Jan. 2000.

[22] Briggs, W. Magnetic calibration by tetrahedral interpolation. In *Proc. NIST-ASME Industrial Virtual Reality Symposium* (Chicago, IL, Nov. 1999), vol. MH-5/MED-9, pp. 27–32.

[23] Brooks, F. P., Ouh-Young, M., Batter, J. J., and Kilpatrick, P. J. Project GROPE – Haptic displays for scientific visualization. In *Proceedings ACM SIGGRAPH Conference* (Dallas, Texas, Aug. 1990), pp. 177–185.

[24] Bruckner, S., and Gröller, E. Volumeshop: An interactive system for direct volume illustration. In *Proceedings IEEE Visualization Conference* (Minneapolis, Minnesota, Oct. 2005), pp. 671–678.

[25] Bryson, S. Measurement and calibration of static distortion of position data from 3D trackers. In *Proceedings SPIE Stereoscopic Displays and Applications Conference* (San Jose, California, Feb. 1992), pp. 244–255.

[26] Bryson, S. *Virtual Reality Applications.* Academic Press, London, United Kingdom, 1995, ch. Approaches to the Successful Design and Implementation of VR Applications, pp. 3–15.

[27] Bryson, S., and Levit, C. The virtual windtunnel: An environment for the exploration of three-dimensional unsteady flows. In *Proceedings IEEE Visualization Conference* (San Diego, California, Oct. 1991), pp. 17–24.

[28] Çavuşoğlu, M. C., Feygin, D., and Tendick, F. Critical study of the mechanical and electrical properties of the PHANToM haptic interface and improvements for high performance control. *Presence: Teleoperators and Virtual Environments 11*, 6 (2002), 555–568.

[29] Cignoni, P., Montani, C., and Scopigno, R. MagicSphere: an insight tool for 3D data visualization. *Computer Graphics Forum 13*, 3 (1994), 317–328.

[30] Cohen, M., and Brodlie, K. Focus and context for volume visualization. In *Proceedings Theory and Practice of Computer Graphics Conference* (Bournemouth, United Kingdom, 2004), pp. 32–39.

[31] Colgate, J. E., and Brown, J. M. Factors affecting the Z-width of a haptic display. In *Proceedings IEEE International Conference on Robotics and Automation* (San Diego, California, May 1994), pp. 3205–3210.

[32] Colgate, J. E., and Brown, J. M. Issues in the haptic display of tool use. In *Proceedings IEEE International Conference on Intelligent Robots and Systems* (Pittsburgh, Pennsylvania, Aug. 1995), pp. 140–145.

[33] Coquillart, S., and Wesche, G. The virtual palette and the virtual remote control panel: A device and and interaction paradigm for the responsive workbench. In *Proceedings IEEE Virtual Reality Conference* (Houston, Texas, Mar. 1999), pp. 213–216.

[34] Cruz-Neira, C., Sandin, D. J., and DeFanti, T. A. Surround-screen projection-based virtual reality: The design and implementation of the CAVE. In *Proceedings ACM SIGGRAPH Conference* (Anaheim, California, Aug. 1993), pp. 135–142.

[35] Cutler, L. D., Fröhlich, B., and Hanrahan, P. Two-handed direct manipulation on the responsive workbench. In *Proceedings ACM Symposium on Interactive 3D Graphics* (Providence, Rhode Island, Apr. 1997), pp. 107–114.

[36] Czernuszenko, M., Sandin, D. J., and DeFanti, T. A. Line of sight method for tracker calibration in projection-based VR systems. In *Proceedings Immersive Projection Technology Workshop* (Ames, Iowa, May 1998).

[37] de Haan, G., Koutek, M., and Post, F. H. Towards intuitive exploration tools for data visualization in VR. In *Proceedings ACM Virtual Reality Software and Technology Conference* (Hong Kong, China, Nov. 2002).

[38] Deering, M. High resolution virtual reality. In *Proceedings ACM SIGGRAPH Conference* (Chicago, Illinois, July 1992), pp. 195–202.

[39] Donald, B. R., and Henle, F. Using haptic vector fields for animation motion control. In *Proceedings IEEE International Conference on Robotics and Automation* (San Francisco, California, Apr. 2000), pp. 3435–3442.

[40] Durbeck, L. J. K., Macias, N. J., Weinstein, D. M., Johnson, C. R., and Hollerbach, J. M. SCIRun haptic display for scientific visualization. In *Proceedings PHANToM Users Group Workshop* (Dedham, Massachusetts, Oct. 1998).

[41] Durlach, N. I., and Mavor, A. S., Eds. *Virtual Reality: Scientific and Technological Challenges*. National Academy Press, Washington, D.C., 1994.

[42] Ebert, D. S., and Shaw, C. D. Minimally immersive flow visualization. *IEEE Transactions on Visualization and Computer Graphics 7*, 4 (2001), 343–350.

[43] Ellis, S. R., Adelstein, B. D., Baumeler, S., Jense, G. J., and Jacoby, R. H. Sensor spatial distortion, visual latency, and update rate effects on 3D tracking in virtual environments. In *Proceedings IEEE Virtual Reality Conference* (Houston, Texas, Mar. 1999), pp. 218–221.

[44] Engel, K., Kraus, M., and Ertl, T. High-quality pre-integrated volume rendering using hardware-accelerated pixel shading. In *Proceedings ACM SIGGRAPH/Eurographics Workshop on Graphics Hardware* (Los Angeles, California, Aug. 2001), pp. 9–16.

[45] Feygin, D., Keehner, M., and Tendick, F. Haptic guidance: Experimental evaluation of a haptic training method for a perceptual motor skill. In *Proceedings IEEE Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (Orlando, Florida, Mar. 2002), pp. 40–47.

[46] Forsberg, A., Kirby, M., Laidlaw, D. H., Karniadakis, G., van Dam, A., and Elion, J. Immersive virtual reality for visualizing flow through an artery. In *Proceedings IEEE Visualization Conference* (Salt Lake City, Utah, Oct. 2000), pp. 457–460.

[47] Forsberg, A. S., Herndon, K. P., and Zeleznik, R. C. Aperture based selection for immersive virtual environments. In *Proceedings ACM User Interface Software and Technology Conference* (Seattle, Washington, Nov. 1996), pp. 95–96.

[48] Forsberg, A. S., LaViola, Jr., J. J., and Zeleznik, R. C. Ergodesk: A framework for two- and three-dimensional interaction at the activedesk. In *Proceedings Immersive Projection Technology Workshop* (Ames, Iowa, May 1998).

[49] Fritz, J. P. Haptic rendering techniques for scientific visualization. Master's thesis, Department of Electrical Engineering, University of Delaware, Newark, Delaware, Dec. 1996.

[50] Fritz, J. P., and Barner, K. E. Design of a haptic graphing system. In *Proceedings RESNA Technology & Disability Conference* (Salt Lake City, Utah, June 1996).

[51] Fröhlich, B., Tramberend, H., Beers, A., Agrawala, M., and Baraff, D. Physically-based manipulation on the responsive workbench. In *Proceedings IEEE Virtual Reality Conference* (New Brunswick, New Jersey, Mar. 2000), pp. 5–12.

[52] Fuhrmann, A. L., and Gröller, M. E. Real-time techniques for 3D flow visualization. In *Proceedings IEEE Visualization Conference* (Research Triangle Park, North Carolina, Oct. 1998), pp. 305–312.

[53] Fuhrmann, A. L., Splechtna, R., and Přikryl, J. Comprehensive calibration and registration procedures for augmented reality. In *Proceedings Eurographics Workshop on Virtual Environments* (Stuttgart, Germany, May 2001), pp. 219–228.

[54] Ghazisaedy, M., Adamczyk, D., Sandin, D. J., Kenyon, R. V., and DeFanti, T. A. Ultrasonic calibration of a magnetic tracker in a virtual reality space. In *Proceedings IEEE Virtual Reality Annual International Symposium* (Research Triangle Park, North Carolina, Mar. 1995), pp. 179–188.

[55] Gottschalk, S., and Hughes, J. F. Autocalibration for virtual environments tracking hardware. In *Proceedings ACM SIGGRAPH Conference* (Anaheim, California, Aug. 1993), pp. 65–72.

[56] Grant, B., Helser, A., and Taylor II, R. M. Adding force display to a stereoscopic head-tracked projection display. In *Proceedings IEEE Virtual Reality Annual International Symposium* (Atlanta, Georgia, Mar. 1998), pp. 81–88.

[57] Grassia, F. S. Practical parameterization of rotations using the exponential map. *Journal of Graphics Tools 3*, 3 (1998), 29–48.

[58] Grosjean, J., and Coquillart, S. Command & control cube: A shortcut paradigm for virtual environments. In *Proceedings Eurographics Workshop on Virtual Environments* (Stuttgart, Germany, May 2001).

[59] Hadwiger, M., Berger, C., and Hauser, H. High-quality two-level volume rendering of segmented data sets on consumer graphics hardware. In *Proceedings IEEE Visualization Conference* (Seattle, Washington, Oct. 2003), pp. 301–308.

[60] Hannaford, B., and Ryu, J.-H. Time domain passivity control of haptic interfaces. *IEEE Transactions on Robotics and Automation 18*, 1 (2002), 1–10.

[61] Harding, C., Loftin, B., and Anderson, A. Visualization and modeling of geoscientific data on the Interactive Workbench. *The Leading Edge 19*, 5 (May 2000), 506–511.

[62] He, T., and Kaufman, A. Virtual input devices for 3D systems. In *Proceedings IEEE Visualization Conference* (San Jose, California, Oct. 1993), pp. 142–148.

[63] Hinckley, K., Pausch, R., Goble, J. C., and Kassell, N. F. Passive real-world interface props for neurosurgical visualization. In *Proceedings ACM SIGCHI Conference* (Boston, Massachusetts, Apr. 1994), pp. 452–458.

[64] Hinckley, K., Tullio, J., Pausch, R., Proffitt, D., and Kassell, N. Usability analysis of 3D rotation techniques. In *Proceedings ACM User Interface Software and Technology Conference* (Banff, Alberta, Oct. 1997), pp. 1–10.

[65] Hirota, K., Hirayam, M., Tanaka, A., and Kaneko, T. Spatial constraint method: A new approach to real-time haptic interaction in virtual environments. *Presence: Teleoperators and Virtual Environments 13*, 3 (2004), 355–370.

[66] Hodges, L. F., and Davis, E. T. Geometric considerations for stereoscopic virtual environments. *Presence: Teleoperators and Virtual Environments 2*, 1 (1993), 34–43.

[67] Hollerbach, J. M., and Wampler, C. W. The calibration index and taxonomy of robot kinematic calibration methods. *International Journal of Robotics Research 15*, 6 (1996), 573–591.

[68] Holloway, R. L. *Registration Errors in Augmented Reality Systems*. PhD thesis, Department of Computer Science, University of North Carolina at Chapel Hill, 1995.

[69] Holloway, R. L. Registration error analysis for augmented reality. *Presence: Teleoperators and Virtual Environments 6*, 4 (1997), 413–432.

[70] Hutchins, M. A constraint equation algebra as a basis for haptic rendering. In *Proceedings PHANToM Users Group Workshop* (Aspen, Colorado, Oct. 2000).

[71] Hutchins, M. Software components for haptic constraints. In *Proceedings SPIE Stereoscopic Displays and Virtual Reality Systems Conference* (San Jose, California, Jan. 2000), pp. 423–432.

[72] Hutchins, M., and Gunn, C. A haptic constraints class library. In *Proceedings PHANToM Users Group Workshop* (Aspen, Colorado, Oct. 1999).

[73] IDELIX Software Inc. Pliable display technology. `http://www.idelix.com/`.

[74] Ikits, M. Coregistration of pose measurement devices using nonlinear total least squares parameter estimation. Technical Report UUCS-00-018, School of Computing, University of Utah, Dec. 2000.

[75] Ikits, M., and Brederson, J. D. *The Visualization Handbook*. Academic Press, Orlando, Florida, 2004, ch. The Visual Haptic Workbench, pp. 431–447.

[76] Ikits, M., Brederson, J. D., Hansen, C. D., and Hollerbach, J. M. An improved calibration framework for electromagnetic tracking devices. In *Proceedings IEEE Virtual Reality Conference* (Yokohama, Japan, Mar. 2001), pp. 63–70.

[77] Ikits, M., Brederson, J. D., Hansen, C. D., and Johnson, C. R. A constraint-based technique for haptic volume exploration. In *Proceedings IEEE Visualization Conference* (Seattle, Washington, Oct. 2003), pp. 263–269.

[78] Ikits, M., and Hansen, C. D. The proxy chain method and its application to scientific visualization. In *Proceedings IEEE Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (Arlington, Virginia, Mar. 2006), pp. 525–532.

[79] Ikits, M., Hansen, C. D., and Johnson, C. R. A comprehensive calibration and registration procedure for the visual haptic workbench. In *Proceedings Eurographics Workshop on Virtual Environments* (Zurich, Switzerland, May 2003), pp. 247–254.

[80] Ikits, M., Kniss, J., Lefohn, A., and Hansen, C. *GPU Gems: Programming Techniques, Tips, and Tricks for Real-Time Graphics.* Addison Wesley, 2004, ch. Volume Rendering Techniques. 667–692.

[81] Immersion, Inc. Medical simulators. `http://www.immersion.com/`.

[82] Infed, F., Brown, S. V., Lee, C. D., Lawrence, D. A., Dougherty, A. M., and Pao, L. Y. Combined visual/haptic rendering modes for scientific visualization. In *Proceedings ASME Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (Nashville, Tennessee, Nov. 1999), pp. 93–99.

[83] Itkowitz, B., Handley, J., and Zhu, W. The OpenHaptics™ toolkit: A library for adding 3D Touch™ navigation and haptics to graphics applications. In *Proceedings World Haptics Conference* (Pisa, Italy, Mar. 2005).

[84] Iwata, H., and Noma, H. Volume haptization. In *Proceedings IEEE Virtual Reality Annual International Symposium* (Seattle, Washington, Sept. 1993), pp. 16–23.

[85] Katz, A. Special rotation vectors – a means for transmitting quaternion information in three components. *Journal of Aircraft 30*, 1 (1993), 148–150.

[86] Keahey, T. A. The generalized detail-in-context problem. In *Proceedings IEEE Symposium on Information Visualization* (Research Triangle Park, North Carolina, Oct. 1998), pp. 44–51.

[87] Kelso, J., Arsenault, L. E., and Kriz, R. D. DIVERSE: A framework for building extensible and reconfigurable device independent virtual environments. In *Proceedings IEEE Virtual Reality Conference* (Orlando, Florida, Mar. 2002), pp. 183–192.

[88] Kim, L., Kyrikou, A., Desbrun, M., and Sukhatme, G. An implicit-based haptic rendering technique. In *Proceedings IEEE International Conference on Intelligent Robots and Systems* (Lausanne, Switzerland, Oct. 2002).

[89] Kindlmann, G. L. The Teem Toolkit. `http://teem.sourceforge.net/`.

[90] Kindlmann, G. L., Weinstein, D. M., and Hart, D. A. Strategies for direct volume rendering of diffusion tensor fields. *IEEE Transactions on Visualization and Computer Graphics 6*, 2 (2000), 124–138.

[91] Kindratenko, V. V. Calibration of electromagnetic tracking devices. *Virtual Reality: Research, Development, and Applications 4* (1999), 139–150.

[92] Kindratenko, V. V., and Bennett, A. J. Evaluation of rotation correction techniques for electromagnetic position tracking systems. In *Proceedings Eurographics Workshop on Virtual Environments* (Amsterdam, The Netherlands, June 2000), pp. 13–22.

[93] Kniss, J., Premože, S., Ikits, M., Lefohn, A., Hansen, C., and Praun, E. Gaussian transfer functions for multi-field volume visualization. In *Proceedings IEEE Visualization Conference* (Seattle, Washington, Oct. 2003), pp. 497–504.

[94] Kniss, J. M., Kindlmann, G. L., and Hansen, C. D. Multidimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics 8*, 3 (2002), 270–285.

[95] Komerska, R., and Ware, C. Haptic task constraints for 3D interaction. In *Proceedings IEEE Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (Los Angeles, California, Mar. 2003), pp. 270–277.

[96] Koutek, M., and Post, F. H. Spring-based manipulation tools for virtual environments. In *Proceedings Eurographics Workshop on Virtual Environments* (Stuttgart, Germany, May 2001).

[97] Koutek, M., and Post, F. H. The responsive workbench simulator: a tool for application development and analysis. In *Proceedings International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision* (Plzeň, Czech Republic, Feb. 2002).

[98] Krüger, J., Schneider, J., and Westermann, R. Clearview: An interactive context preserving hotspot visualization technique. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 941–948.

[99] Krüger, J., and Westermann, R. Acceleration techniques for GPU-based volume rendering. In *Proceedings IEEE Visualization Conference* (Seattle, Washington, Oct. 2003), pp. 287–292.

[100] Kuipers, J. B. SPASYN – an electromagnetic relative position and orientation tracking system. *IEEE Transactions on Instrumentation and Measurement 29*, 4 (1980), 462–466.

[101] Kuipers, J. B. *Quaternions and Rotation Sequences*. Princeton University Press, Princeton, NJ, 1999.

[102] Laidlaw, D. H., Kirby, R. M., Jackson, C. D., Davidson, J. S., Miller, T. S., da Silva, M., Warren, W. H., and Tarr, M. J. Comparing 2D vector field visualization methods: A user study. *IEEE Transactions on Visualization and Computer Graphics 11*, 1 (2005), 59–70.

[103] LaMar, E., Hamann, B., and Joy, K. I. A magnification lens for interactive volume visualization. In *Proceedings Pacific Conference on Computer Graphics and Applications* (Tokyo, Japan, Oct. 2001), pp. 223–232.

[104] LaMar, E. C., Hamann, B., and Joy, K. I. Multiresolution techniques for interactive texture-based volume visualization. In *Proceedings IEEE Visualization Conference* (San Francisco, California, Oct. 1999), pp. 355–362.

[105] Lawrence, D. A., Lee, C. D., Pao, L. Y., and Novoselov, R. Y. Shock and vortex visualization using a combined visual/haptic interface. In *Proceedings IEEE Visualization Conference* (Salt Lake City, Utah, Oct. 2000), pp. 131–137.

[106] Lefohn, A., Kniss, J. M., Strzodka, R., Sengupta, S., and Owens, J. D. Glift: Generic GPU data structures. *ACM Transactions on Graphics 25*, 1 (2006), 60–99.

[107] Lefohn, A. E., Kniss, J. M., Hansen, C. D., and Whitaker, R. T. A streaming narrow-band algorithm: Interactive deformation and visualization of level sets. *IEEE Transactions on Visualization and Computer Graphics 10*, 4 (2004). 422–433.

[108] Li, X., and Shen, H.-W. Time-critical multiresolution volume rendering using 3D texture mapping hardware. In *Proceedings IEEE Volume Visualization and Graphics Symposium* (Boston, Massachusetts, Oct. 2002), pp. 29–36.

[109] Lindeman, R. W., Sibert, J. L., and Templeman, J. N. The effect of 3D widget representation and simulated surface constraints on interaction in virtual environments. In *Proceedings IEEE Virtual Reality Conference* (Yokohama, Japan, Mar. 2001).

[110] Livingston, M. A., and State, A. Magnetic tracker calibration for improved augmented reality registration. *Presence: Teleoperators and Virtual Environments 6*, 5 (1997), 532–546.

[111] Logitech, Inc. Game controllers. `http://www.logitech.com/`.

[112] Lundin, K. Natural haptic feedback from volumetric density data. Master's thesis, Linköping University, Sweden, Dec. 2001.

[113] Lundin, K., Gudmundsson, B., and Ynnerman, A. General proxy-based haptics for volume visualization. In *Proceedings World Haptics Conference* (Pisa, Italy, Mar. 2005).

[114] Lundin, K., Ynnerman, A., and Gudmundsson, B. Proxy-based haptic feedback from volumetric density data. In *Proceedings Eurohaptics Conference* (Edinburgh, United Kingdom, July 2002).

[115] Maciejewski, R., Choi, S., Ebert, D. S., and Tan, H. Z. Multi-model perceptualization of volumetric data and its application to molecular docking. In *Proceedings World Haptics Conference* (Pisa, Italy, Mar. 2005).

[116] Mascarenhas, A., Ehmann, S., Gregory, A., Lin, M., and Manocha, D. Six degree-of-freedom haptic visualization. In *Proceedings Touch in Virtual Environments Conference* (Los Angeles, CA, Feb. 2001).

[117] Massie, T. H. Design of a three degree of freedom force-reflecting haptic interface. Bachelor's thesis, Massachusetts Institute of Technology, 1993.

[118] Massie, T. H. Initial haptic explorations with the PHANToM: Virtual touch through point interaction. Master's thesis, Massachusetts Institute of Technology, 1996.

[119] Maxwell, S., and Delaney, H. *Designing Experiments and Analyzing Data: A Model Comparison Perspective.* Wadsworth Publishing Company, Belmont, California, 1990.

[120] McLaughlin, J. P., and Orenstein, B. J. Haptic rendering of 3D seismic data. In *Proceedings PHANToM Users Group Workshop* (Dedham, Massachusetts, Oct. 1997).

[121] Méndez, E., Yoshida, S., Noma, H., Lindeman, R. W., Yanagida, Y., Masaki, S., and Hosaka, K. A haptic-assisted guidance system for navigating volumetric data sets. In *Proceedings World Haptics Conference* (Pisa, Italy, Mar. 2005).

[122] Meyer, M., and Barr, A. H. ALCOVE: Design and implementation of an object-centric virtual environment. In *Proceedings IEEE Virtual Reality Conference* (Houston, Texas, Mar. 1999), pp. 46–52.

[123] Meyer, T., and Globus, A. Direct manipulation of isosurfaces and cutting planes in virtual environments. Technical Report CS-93-54, Department of Computer Science, Brown University, Dec. 1993.

[124] Miller, T., and Zeleznik, R. C. The design of 3D haptic widgets. In *Proceedings ACM Symposium on Interactive 3D Graphics* (Atlanta, Georgia, Apr. 1999), pp. 97–102.

[125] Munzner, T., Guimbretiere, F., Tasiran, S., Zhang, L., and Zhou, Y. TreeJuxtaposer: Scalable tree comparison using focus+context with guaranteed visibility. *ACM Transactions on Graphics 22*, 3 (2003), 453–462.

[126] Nahvi, A., and Hollerbach, J. M. Display of friction in virtual environments based on human finger pad characteristics. In *Proceedings ASME Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (Anaheim, California, Nov. 1998), pp. 179–184.

[127] Nielsen, P. M. F., LeGrice, I. J., Smaill, B. H., and Hunter, P. J. Mathematical model of geometry and fibrous structure of the heart. *American Journal of Physiology 260*, 29 (1991), H1365–H1378.

[128] Nixon, M. A., McCallum, B. C., Fright, W. R., and Price, N. B. The effects of metals and interfering fields on electromagnetic trackers. *Presence: Teleoperators and Virtual Environments 7*, 2 (1998), 204–218.

[129] Northern Digital, Inc. `http://www.ndigital.com/`.

[130] Novoselov, R. Y., Lawrence, D. A., and Pao, L. Y. Haptic rendering of data on unstructured tetrahedral grids. In *Proceedings IEEE Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (Orlando, Florida, Mar. 2002), pp. 193–200.

[131] Oakley, I., Adams, A., Brewster, S., and Gray, P. Guildelines for the design of haptic widgets. In *Proceedings Human-Computer Interaction Conference* (London, United Kingdom, Sept. 2002).

[132] Obeysekare, U., Williams, C., Durbin, J., Rosenblum, L., Rosenberg, R., Grinstein, F., Ramamurti, R., Landsberg, A., and Sandberg, W. Virtual workbench – A non-immersive virtual environment for visualizing and interacting

with 3D objects for scientific visualization. In *Proceedings IEEE Visualization Conference* (San Francisco, California, Oct. 1996), pp. 345–349.

[133] Okamura, A. M., Dennerlein, J. T., and Howe, R. D. Vibration feedback models for virtual environments. In *Proceedings IEEE International Conference on Robotics and Automation* (Leuven, Belgium, May 1998), pp. 2485–2490.

[134] Otaduy, M. A., and Lin, M. C. User-centric viewpoint computation for haptic exploration and manipulation. In *Proceedings IEEE Visualization Conference* (San Diego, California, Oct. 2001), pp. 311–318.

[135] Paljic, A., Burkhardt, J.-M., and Coquillart, S. A study of distance of manipulation on the responsive workbench. In *Proceedings Immersive Projection Technology Workshop* (Orlando, Florida, Mar. 2002).

[136] Pao, L. Y., and Lawrence, D. A. Synergistic visual/haptic computer interfaces. In *Proceedings Japan/USA/Vietnam Workshop on Research and Education in Systems, Computation, and Control Engineering* (Hanoi, Vietnam, May 1998), pp. 155–162.

[137] Passmore, P., Nielsen, C., Cosh, W., and Darzi, A. Effects of viewing and orientation on path following in a medical teleoperation task. In *Proceedings IEEE Virtual Reality Conference* (Yokohama, Japan, Mar. 2001), pp. 209–215.

[138] Peterson, G. L. Concurrent reading while writing. *ACM Transactions on Programming Languages and Systems 5*, 1 (1983), 46–55.

[139] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. *Numerical Recipes in C*, 2nd ed. Cambridge University Press, Cambridge, 1992.

[140] Raab, F. H., Blood, E. B., Steiner, T. O., and Jones, H. R. Magnetic position and orientation tracking system. *IEEE Transactions on Aerospace and Electronic Systems 15*, 5 (1979), 709–718.

[141] Rajlich, P. J. An object oriented approach to developing visualization tools portable across desktop and virtual environments. Master's thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana-Champaign, Illinois, May 1998.

[142] Reinig, K., Tracy, R., Gilmore, H., and Mahalik, T. Some calibration information for a PHANToM 1.5A. In *Proceedings PHANToM Users Group Workshop* (Dedham, Massachusetts, Oct. 1997).

[143] Robinett, W., and Rolland, J. P. A computational model for the stereoscopic optics of a head-mounted display. *Presence: Teleoperators and Virtual Environments 1*, 1 (1991), 45–62.

[144] Roettger, S., and Ertl, T. A two-step approach for interactive pre-integrated volume rendering of unstructured grids. In *Proceedings IEEE Volume Visualization and Graphics Symposium* (Boston, Massachusetts, Oct. 2002), pp. 23–28.

[145] Roettger, S., Guethe, S., Weiskopf, D., Ertl, T., and Strasser, W. Smart hardware-accelerated volume rendering. In *Proceedings Joint Eurographics/IEEE TVCG Symposium on Visualization* (Grenoble, France, May 2003), pp. 231–238.

[146] Rolland, J. P., Ariely, D., and Gibson, W. Towards quantifying depth and size perception in virtual environments. *Presence: Teleoperators and Virtual Environments 4*, 1 (1995), 24–49.

[147] Ropinksi, T., and Hinrichs, K. Real-time rendering of 3D magic lenses having arbitrary convex shapes. In *Proceedings International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision* (Plzeň, Czech Republic, Feb. 2004).

[148] Rosenberg, L. B. Virtual fixtures: Perceptual tools for telerobotic manipulation. In *Proceedings IEEE Virtual Reality Annual International Symposium* (Seattle, Washington, Sept. 1993), pp. 76–82.

[149] Ruspini, D. C., Kolarov, K., and Khatib, O. The haptic display of complex graphical environments. In *Proceedings ACM SIGGRAPH Conference* (Los Angeles, California, Aug. 1997), pp. 345–352.

[150] Salisbury, J. K., and Tarr, C. Haptic rendering of surfaces defined by implicit functions. In *Proceedings ASME Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (Dallas, Texas, Nov. 1997), pp. 61–67.

[151] Salisbury, K., Brock, D., Massie, T., Swarup, N., and Zilles, C. Haptic rendering: Programming touch interaction with virtual objects. In *Proceedings ACM Symposium on Interactive 3D Graphics* (Monterey, California, Apr. 1995), pp. 123–130.

[152] Sankaranarayanan, G., Devarajan, V., Eberhart, R., and Jones, D. B. Adaptive hybrid interpolation techniques for direct haptic rendering of isosurfaces. In *Proceedings Medicine Meets Virtual Reality Conference* (Newport Beach, California, Jan. 2002), pp. 448–454.

[153] Sayers, C. P., and Paul, R. P. An operator interface for teleprogramming employing synthetic fixtures. *Presence: Teleoperators and Virtual Environments 3*, 4 (1994), 309–320.

[154] Schmalstieg, D., Encarnação, L. M., and Szalavári, Z. Using transparent props for interaction with the virtual table. In *Proceedings ACM Symposium on Interactive 3D Graphics* (Atlanta, Georgia, Apr. 1999), pp. 147–153.

[155] Schroeder, W., Martin, K., and Lorensen, W. *The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics*. Prentice Hall, Upper Saddle River, New Jersey, 1996.

[156] Schulze, J. P., Wössner, U., Walz, S. P., and Lang, U. Volume rendering in a virtual environment. In *Proceedings Immersive Projection Technology Workshop* (Stuttgart, Germany, June 2001), pp. 187–198.

[157] Schwetlick, H., and Tiller, V. Numerical methods for estimating parameters in nonlinear models with errors in the variables. *Technometrics 27*, 1 (1985), 17–24.

[158] Seeger, A., Henderson, A., Pelli, G. L., Hollins, M., and Taylor II, R. M. Haptic display of multiple scalar fields on a surface. In *Proceedings Workshop on New Paradigms in Information Visualization and Manipulation* (Washington, D.C., Nov. 2000), pp. 33–38.

[159] SensAble Technologies, Inc. Freeform system. `http://www.sensable.com/`.

[160] Serra, L., Hern, N., Choon, C. B., and Poston, T. Interactive vessel tracing in volume data. In *Proceedings ACM Symposium on Interactive 3D Graphics* (Providence, Rhode Island, Apr. 1997), pp. 131–137.

[161] Serra, L., Hern, N., Guan, C. G., Lee, E., Lee, Y. H., T.T., Y., Chan, C., and Kockro, R. A. An interface for precise and comfortable 3D work with volumetric medical datasets. In *Proceedings Medicine Meets Virtual Reality Conference* (San Francisco, California, Jan. 1999).

[162] Serra, L., Poston, T., Hern, N., Choon, C. B., and Waterworth, J. A. Interaction techniques for a virtual workspace. In *Proceedings ACM Virtual Reality Software and Technology Conference* (Chiba, Japan, Nov. 1995), pp. 79–90.

[163] Shaw, C. D., Hall, J. A., Ebert, D. S., and Roberts, D. A. Interactive lens visualization techniques. In *Proceedings IEEE Visualization Conference* (San Francisco, California, Oct. 1999), pp. 155–159.

[164] Srinivasan, M. A., and Basdogan, C. Haptics in virtual environments: Taxonomy, research status, and challenges. *Computer and Graphics 21*, 4 (1997), 393–404.

[165] Stevenson, D. R., Smith, K. A., McLaughlin, J. P., Gunn, C. J., Veldkamp, J. P., and Dixon, M. J. Haptic workbench: A multisensory virtual environment. In *Proceedings SPIE Stereoscopic Displays and Virtual Reality Systems Conference* (San Jose, California, Jan. 1999), pp. 356–366.

[166] Strauss, P. S., and Carey, R. An object-oriented 3D graphics toolkit. In *Proceedings ACM SIGGRAPH Conference* (Chicago, Illinois, July 1992), pp. 341–349.

[167] Summers, V. A., Booth, K. S., Calvert, T., Graham, E., and MacKenzie, C. L. Calibration for augmented reality experimental testbeds. In *Proceedings ACM Symposium on Interactive 3D Graphics* (Atlanta, Georgia, Apr. 1999), pp. 155–162.

[168] Tuceryan, M., Greer, D. S., Whitaker, R. T., Breen, D. E., Rose, E., Ahlers, K. H., and Crampton, C. Calibration requirements and procedures for a monitor-based augmented reality system. *IEEE Transactions on Visualization and Computer Graphics 1*, 3 (1995), 255–273.

[169] Turk, G., and Banks, D. Image-guided streamline placement. In *Proceedings ACM SIGGRAPH Conference* (New Orleans, Louisiana, Aug. 1996), pp. 453–460.

[170] Tzeng, F.-Y., Lum, E. B., and Ma, K.-L. A novel interface for higher-dimensional classification of volume data. In *Proceedings IEEE Visualization Conference* (Seattle, Washington, Oct. 2003), pp. 505–512.

[171] van Dam, A., Forsberg, A. S., Laidlaw, D. H., LaViola, Jr., J. J., and Simpson, R. M. Immersive VR for scientific visualization: A progress report. *IEEE Computer Graphics and Applications 20*, 6 (2000), 26–52.

[172] van Reimersdahl, T., Bley, F., Kuhlen, T., and Bischof, C. Haptic rendering techniques for the interactive exploration of CFD datasets in virtual environments. In *Proceedings Eurographics Workshop on Virtual Environments* (Zurich, Switzerland, May 2003), pp. 241–246.

[173] Vidholm, E., and Nyström, I. A haptic interaction technique for volume images based on gradient diffusion. In *Proceedings World Haptics Conference* (Pisa, Italy, Mar. 2005).

[174] Viega, J., Conway, M. J., Williams, G., and Pausch, R. 3D magic lenses. In *Proceedings ACM User Interface Software and Technology Conference* (Seattle, Washington, Nov. 1996), pp. 51–58.

[175] von Wiegand, T. E., Schloerb, D. W., and Sachtler, W. L. Virtual workbench: Near-field virtual environment system with applications. *Presence: Teleoperators and Virtual Environments 8*, 5 (1999), 492–519.

[176] Wall, S. A., and Harwin, W. S. Quantification of the effects of haptic feedback during a motor skills task in a simulated environment. In *Proceedings PHANToM Users Research Symposium* (Zurich, Switzerland, July 2000).

[177] Wall, S. A., Paynter, K., Shillito, A. M., Wright, M., and Scali, S. The effect of haptic feedback and stereo graphics in a 3D target acquisition task. In *Proceedings Eurohaptics Conference* (Edinburgh, United Kingdom, July 2002), pp. 23–29.

[178] Wang, L., Zhao, Y., Mueller, K., and Kaufman, A. The magic volume lens: An interactive focus+context technique for volume rendering. In *Proceedings IEEE Visualization Conference* (Minneapolis, Minnesota, Oct. 2005), pp. 367–374.

[179] Wartell, Z., Hodges, L. F., and Ribarsky, W. The analytic distortion induced by false-eye separation in head-tracked stereoscopic displays. Technical Report GIT-GVU-99-01, GVU Center, Georgia Institute of Technology, Jan. 1999.

[180] Wartell, Z., Hodges, L. F., and Ribarsky, W. A geometric comparison of algorithms for fusion control in stereoscopic HTDs. *IEEE Transactions on Visualization and Computer Graphics 8*, 2 (2002), 129–143.

[181] Weiler, M., Kraus, M., Merz, M., and Ertl, T. Hardware-based ray casting for tetrahedral meshes. In *Proceedings IEEE Visualization Conference* (Seattle, Washington, Oct. 2003), pp. 333–340.

[182] Weiler, M., Westermann, R., Hansen, C., Zimmerman, K., and Ertl, T. Level-of-detail volume rendering via 3D textures. In *Proceedings IEEE Volume Visualization and Graphics Symposium* (Salt Lake City, Utah, Oct. 2000), pp. 7–13.

[183] Weiskopf, D., Engel, K., and Ertl, T. Interactive clipping techniques for texture-based volume visualization and volume shading. *IEEE Transactions on Visualization and Computer Graphics 9*, 3 (2003), 298–312.

[184] Wohlfahrter, W., Encarnação, L. M., and Schmalstieg, D. Interactive volume exploration on the studydesk. In *Proceedings Immersive Projection Technology Workshop* (Ames, Iowa, June 2000).

[185] Woo, M., Neider, J., Davis, T., and Shreiner, D. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 1.2.* Addison Wesley, Reading, Massachusetts, 1999.

[186] Yi, D., and Hayward, V. Augmenting computer graphics with haptics for the visualization of vessel networks. In *Proceedings Pacific Conference on Computer Graphics and Applications* (Beijing, China, Oct. 2002), pp. 375–384.

[187] Zachmann, G. Distortion correction of magnetic fields for position tracking. In *Proceedings Computer Graphics International Conference* (Hasselt, Belgium, June 1997), pp. 213–220.

[188] Zhang, S., Ç. Demiralp, Keefe, D., DaSilva, M., Laidlaw, D. H., Greenberg, B. D., Basser, P., Pierpaoli, C., Chiocca, E., and Deisboeck, T. An immersive virtual environment for DT-MRI volume visualization applications: A case study. In *Proceedings IEEE Visualization Conference* (San Diego, California, Oct. 2001), pp. 437–440.

[189] Zhou, J., Hinz, M., and Tönnies, K. D. Focal region-guided feature-based volume rendering. In *Proceedings International Symposium on 3D Data Processing, Visualization and Transmission* (Padova, Italy, June 2002), pp. 87–90.

[190] Zilles, C. B., and Salisbury, J. K. A constraint-based god-object method for haptic display. In *Proceedings IEEE International Conference on Intelligent Robots and Systems* (Pittsburgh, Pennsylvania, Aug. 1995), pp. 146–151.

[191] Zöckler, M., Stalling, D., and Hege, H. Interactive visualization of 3D-vector fields using illuminated streamlines. In *Proceedings IEEE Visualization Conference* (San Francisco, California, Oct. 1996), pp. 107–113.