Thesis for the Degree of Doctor of Philosophy

3D Occlusion Management and Causality Visualization

Niklas Elmqvist

CHALMERS | GÖTEBORG UNIVERSITY

Department of Computer Science and Engineering
Chalmers University of Technology
and Göteborg University
412 96 Göteborg, Sweden

Göteborg, November 2006

3D Occlusion Management and Causality Visualization
Niklas Elmqvist
ISBN 91-7291-869-1

Doktorsavhandlingar vid Chalmers tekniska högskola
Ny serie nr 2550
ISSN 0346-718X

Technical Report no. 25D
School of Computer Science and Engineering

Department of Computer Science and Engineering
Chalmers University of Technology and Göteborg University
412 96 Göteborg, Sweden
Telephone +46 (0)31-772 1000

Cover: The BalloonProbe 3D occlusion management technique.

Chalmers Reproservice
Göteborg, Sweden 2006

3D Occlusion Management and Causality Visualization

Niklas Elmqvist
Department of Computer Science and Engineering
Chalmers University of Technology

Abstract

This thesis is split into two parts: one part dealing with the management of occlusion in 3D environments, the other with the visualization of causal relations. Both of these parts fall within the general framework of visualization—the graphical representation of data (abstract or concrete) with the purpose of amplifying cognition—but they do so in different ways.

3D occlusion management, on the one hand, is a basic approach to augmenting three-dimensional visualizations with a set of techniques for reducing (or even eliminating) the effect of inter-object occlusion in the environment. We present four different such techniques in the thesis, each utilizing a different solution space to achieve the effect: image space for dynamic transparency, view space for view projection animation, object space for interactive 3D distortion, and temporal space for our approach to 3D navigation guidance. Each technique is orthogonal to the others, and each has been verified empirically through formal user experiments to promote significantly more efficiency and accuracy for users solving representative visual perception task in 3D environments than standard navigation controls such as flying and walking.

Causality visualization, on the other hand, is a specific class of visualization techniques designed to make complex chains of causal dependencies, or cause-and-effect relations, visible and understandable. A core information visualization problem, the techniques described in this part of the thesis are examples of growing geometry, a subset of methods based on mapping the time parameter to the size of geometrical primitives such as squares and polygons. Consequently, the techniques are called Growing Squares and Growing Polygons, respectively, and utilize color, texture, and animation to visualize causality from application areas such as distributed systems, social networks, and mathematics. These, too, have been empirically shown to be superior to traditional time-space diagrams for representing causal relations. The CiteWiz system serves as a concrete example of how to apply causality visualization to a real dataset, in this case scientific citation data.

Keywords: *occlusion management, occlusion reduction, visualization, information visualization, interaction techniques, causality, causal relations*

List of Included Papers and Reports

This thesis is based on the following publications and reports:

1. Elmqvist, N., Tsigas, P. A Taxonomy of 3D Occlusion Management Techniques. In *Proceedings of the IEEE Conference on Virtual Reality 2007*, to appear.

2. Elmqvist, N., Tsigas, P. View Projection Animation for Occlusion Reduction. In *Proceedings of the ACM Conference on Advanced Visual Interfaces 2006*, pp. 471–475.

3. Elmqvist, N. BalloonProbe: Reducing Occlusion in 3D using Interactive Space Distortion. In *Proceedings of the ACM Symposium on Virtual Reality Software & Technology 2005*, pp. 134–137.

4. Elmqvist, N., Tudoreanu, M. E. Evaluating the Effectiveness of Occlusion Reduction Techniques for 3D Virtual Environments. In *Proceedings of the ACM Symposium on Virtual Reality Software & Technology 2006*, pp. 9–18.

5. Assarsson, U., Elmqvist, N., Tsigas, P. Image-Space Dynamic Transparency for Improved Object Discovery in 3D Environments. Technical report CS:2006-10, Chalmers University of Technology, Göteborg (2006).

6. Elmqvist, N., Tsigas, P. On Navigation Guidance for Exploration of 3D Environments. Technical report CS:2006-19, Chalmers University of Technology, Göteborg (2006).

7. Elmqvist, N., Tsigas, P. Animated Visualization of Causal Relations Through Growing 2D Geometry. In *Information Visualization*, Vol. 3 (2004) No. 3, pp. 154–172 (Special Issue of Selected Papers from the ACM Symposium on Software Visualisation 2003), Palgrave Macmillan.

8. Elmqvist, N., Tsigas, P. Growing Squares: Animated Visualization of Causal Relations. In *Proceedings of the ACM Symposium on Software Visualization 2003*, pp. 17–26.

9. Elmqvist, N., Tsigas, P. Causality Visualization Using Animated Growing Polygons. In *Proceedings of the IEEE Symposium on Information Visualization 2003*, pp. 189–196.

10. Elmqvist, N., Tsigas, P. CiteWiz: A Tool for the Visualization of Scientific Citation Networks. Technical report CS:2004-05, Chalmers University of Technology, Göteborg (2004).

Preface

Man's mind, once stretched by a new idea,
never regains its original dimensions.
– Oliver Wendell Holmes (1809–1894)

One of my favorite non-fiction books, *Writer's INC*, claims that "writing is mind traveling, destination unknown." I have always liked this statement and used it as guidance for my own writing for a long time. However, in recent years, I have come to see a greater truth. Since so much of research is made up of writing, it is clear to me that research, too, is mind traveling, destination unknown.

My destination certainly was unknown when I set out on this journey, more than five years ago. The very fact that I found myself at the beginning of it was also somewhat of a fluke, one I can only thank my supervisor for. That is another story. Regardless, there I was, fresh out of my undergraduate studies and beginning to warm up to the idea of doing research in computer science. The question was about what, exactly?

That it would be human-computer interaction was clear from the start. I have always had a fascination in that humble layer between man and machine. It may not look much to the world, almost inconsequential, but this is where the magic happens. This is where the battle for technology acceptance is won or lost. Working in this field requires a combination of both theory and practice, knowledge of both technology and human nature. Any part of these two duals on its own is useless. Only together do they make sense.

In the end, my path started off in the exciting field of software visualization and then meandered along into information visualization and later to the more general research area of visualization. During the course of my studies, I have done work on a wide range of subjects, from computer security to Augmented and Virtual Reality. It has been an exhilarating ride.

It is now five years later. I am writing this as a break away from feverish last-minute coding and data collection for my doctoral thesis, the one you are holding in your hands. The destination certainly was unknown from the start, but now that I am near it, I know it, and I am not unhappy about where this great journey has taken me. Yes, research is mind traveling, but even if the destination may be unknown, what does that matter? It is, to borrow a tired old phrase, the way there that counts.

Acknowledgments

If you are anything like me, the acknowledgments section is one of the first things you read in a thesis, if only because this is where the author can show something of a human face in this otherwise most scientific of publications. Never one to disappoint, I will utilize this chance for unscientific chitchat to the fullest.

My first and most sincere acknowledgment must go to my supervisor, Philippas Tsigas. Without his intervention, I would probably never have put foot on the noble path of science, and without his help, I certainly would never have reached this far. Philippas has given meaning to the age-old practice of master and apprentice, and now that I am about to become a journeyman, I can begin to fully appreciate his work. Thank you.

I am very happy and honored to have Dr. Doug Bowman of Virginia Polytechnic Institute as my opponent. Furthermore, my gratitude to the members of my grading committee: Dr. Stephan Diehl, Dr. Morten Fjeld, Dr. Lars Hallnäs, and Dr. Diane Sonnenwald.

I would also like to thank the past and present members of my PhD supervision committee—Henrik Ahlberg, Fang Chen, Ralph Schroeder, and Björn von Sydow—and my examiner—David Sands. Your comments and feedback have given me much-needed perspective on my work when I was far too involved in it to see clearly myself. Special thanks go out to Henrik, who introduced me to the idea of doing research in the first place.

Research is always done in a context, never in the isolation of an ivory tower, and my context has primarily been the Distributed Computing and Systems research group at the Department of Computer Science and Engineering at Chalmers: Daniel Cederman, Anders Gidenstam, Phuong Hoai Ha, Boris Koldehofe, Andreas Larsson, Marina Papatriantafilou, Elad Schiller, Håkan Sundell, and Yi Zhang. Thank you also to John Stasko and his Information Interfaces research group at Georgia Tech in Atlanta, USA, for serving as my context and support for three of the most intense months of my life. Overall, I am grateful to the whole department at Chalmers for promoting an atmosphere that helps you achieve your best, yet still provides a friendly and relaxed working environment.

Finally, I would like to extend my deepest gratitude to my friends and family. Special thanks to my father, Lars-Gunnar, for always taking an interest in my work and constantly urging me onto greater achievements, and to my mother, Anita, for always being supportive and taking care of me the few times each year I come home. Thanks to my brother, Jonas, for being my skiing partner during the holidays and for giving me the kind of friendship that you always know is there, come rain or high water. And to my beloved Helene, my muse and companion—mere words cannot express my feelings, so I will not even try.

Niklas Elmqvist
Göteborg, November 2006

Contents

Chapter 1

Overview

It is difficult to describe the increasing information flow in the society of today without resorting to clichés and superlatives. According to the study "How much information?" [95, 96], conducted at University of California Berkeley in the United States in 2000 and 2003, the world produces around 5 exabytes[1] (between 1 and 2 exabytes in 2000) of new information every year stored on print, film, magnetic and optical storage media. Only .003% of this comprises printed documents. For comparison, five exabytes is equivalent to all words ever spoken by human beings.

Clearly, we have long since reached a point where it is physically impossible for a single human being to appropriate even a small fraction of this amount of information. At the same time, it is also obvious that being able to harness all this readily available information could yield tremendous advantages, especially given that such a large part of the information is now available in a digital form that is tractable to analysis by computers.

However, the question is how do we analyze this information? And if we manage to find an answer to that question, how do we then present the results from our analysis in such a way that our limited human minds can understand them?

This thesis is about visualization, one approach to solving these two questions. Visualization is a research area concerned with the graphical representation of information for the purpose of aiding cognition in a human user. In other words, visualization researchers spend their time thinking up ways to draw pictures (or rather, have the computer draw pictures, many of them moving) that encode the information we are trying to portray in a manner that helps the user more easily understand it. We all know that a picture says more than a thousand words; by inference, we might deduce that this could be a way to solve the information overload problem outlined above. Perhaps we can draw a picture, or a few pictures, that somehow captures and represents all of those five exabytes

[1]An exabyte is a billion gigabytes, or 10^{18} bytes.

of data we produce every year?

Visualization is by its very nature problem-oriented, each of the techniques in the area more or less tailored for a specific kind of information and a specific kind of problem domain. That means that a single visualization technique is unlikely to ever become the "be-all end-all" of data representation. This is generally due to the tradeoff between abstraction and acceptance: abstract representations, with characters and words being the ultimate in abstraction, can represent very complex ideas, yet are difficult to decode and understand readily; more concrete visual representations have less expressive power, but are more easily understood by the user. In visualization, we typically aim to move away from the very abstract and towards the more concrete. We use pictures and icons instead of characters and symbols, metaphors and ideas instead of tangible concepts to represent information. Where you draw the line between the abstract and the concrete is a very complex matter. Suffice to say is that visualization researchers usually target a specific type of data in order to make their visualizations more acceptable and to leverage our inherent visual cognition capabilities.

In this thesis, that particular target is a tiny portion of all the data described above, the portion of the data that deals with either 3D environments or causal relations. More specifically, our work here focuses on two parts: helping users navigate in a computer-generated three-dimensional environment, as well as understanding complex chains of causality, i.e. essentially the concepts of cause-and-effect. Due to the great discrepancy of these two topics, the thesis has been split into two main parts.

Part I deals with 3D occlusion management, a general approach to handle the occlusion effect in 3D environments in order to improve visual perception and cognition of the environment. We begin by defining the general problem space of 3D occlusion and describing a taxonomy of the design space for occlusion management techniques. We then present our different explorations into occlusion management, essentially constituting a toolbox of orthogonal techniques designed to help users more easily view and understand a 3D world.

Part II, on the other hand, represents our second main research effort, the visualization of causal relations. As before, we start this part of the thesis with a general introduction to the area and a review of the existing work. We then go on to describe our two visualization techniques based on growing geometry: Growing Squares and Growing Polygons. We conclude this part with a case study showing the application of causality visualization to bibliographic visualization of scientific citation networks.

Before we launch ourselves into these two topics, however, this chapter will give the basic introduction to visualization and information visualization that is common to both of the two parts. At the end of this chapter, you will also find a summary of contributions of the work contained in the thesis.

1.1 Visualization

The field of *visualization* is concerned with the graphical representation of any kind of data with the purpose of aiding a human user in creating a *mental model* of it. Most research in the area focuses on developing effective representations that allow the user to discover hidden information and interpret the data in new and more efficient ways. These representations are generally called *visualization techniques*, and the mental process they make use of is referred to as *external cognition* [122], or what Norman [107, 108] calls "knowledge in the world."

We define *cognition* as the acquisition or use of knowledge [22], i.e. the process of building a mental model by retrieving meaningful information from raw data.[2] *External cognition* is then cognition aided by the external world, including the interaction between internal and external representations (i.e. the creation of the mental model in the user's head), and is manifest in a wide array of human artifacts ranging from slider rules and maps to diagrams and post-it notes; all examples of real-world aids that amplify cognition. As Norman [108] notes, "it is things that make us smart." Imagine performing a multiplication of two three-digit numbers in your head without the use of pen and paper, and most people will agree that this statement is true.

However, external cognition is a very broad concept, and involves as disparate disciplines as information design and data graphics. To put the field of visualization into perspective, we will use the following definition in this thesis [22]:

> **Visualization:** The use of computer-supported, interactive visual representations of data to amplify cognition.

The data sets involved in this task are often sufficiently huge and complex that they yield next to no information when presented in textual or tabular form. Therefore, visualization may often be the only alternative for a human user to be able to understand and think about the data effectively. In fact, as Card et al. [22] remark, the ubiquity of visual metaphors in describing cognitive processes hints at a strong interrelationship between what we see and what we think: to understand something is called "seeing" it, we try to make our ideas "clear", to bring them into "focus". Herein lies the case for visualization.

Visualization is a very general research area with close ties to the fields of human-computer interaction as well as computer graphics. Today, the field also contains a number of subfields, including specifically *information visualization* and *scientific visualization*. The former deals with abstract data lacking a natural visual mapping, such as document databases, complex hierarchies, and text, whereas the latter is concerned with the representation of data that has some kind of physical or spatial mapping, such as air flow around a car, temperature

[2]It is important to make a clear distinction between *data* and *information*; users wish to derive information from data to gain insight into it, to be informed by the data.

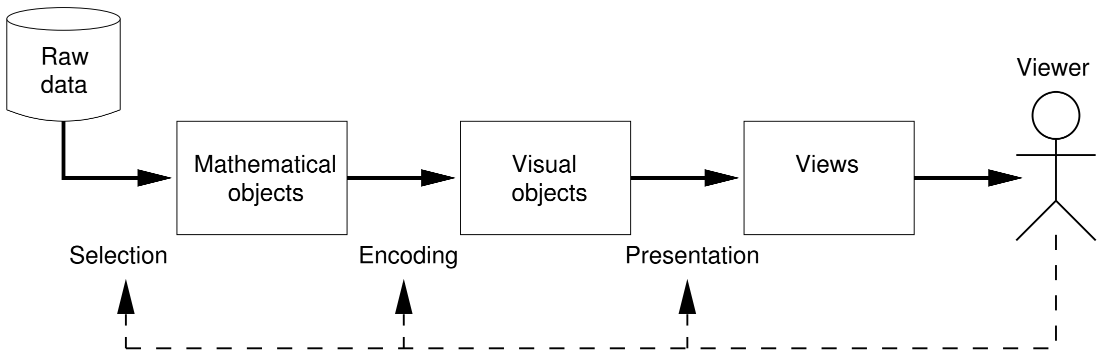


Figure 1.1: A general model for information visualization.

readings in the oceans at various points around the globe, or sound levels in the immediate neighborhood of an airport.

In this section, we will begin by giving an outline of a general visualization model describing the flow and transformation of data from a raw, unstructured form to a visual object suitable for a human viewer. We will describe the concept of visual structures in detail, and view transformations acting upon these. We then give a definition of visualization techniques. The section following that gives more details on information visualization.

1.1.1 Visualization Model

Figure 1.1 presents a general model for visualization showing the flow from raw data to a mental model created by the viewer and the transitions between the various intermediate stages of representation. This model can be applied to any visualization subfield. We describe these stages in the following text.

Selection

The process of *selection* converts raw data of some idiosyncratic format to a mathematical object suitable for encoding into a visual form in the next step of the visualization pipeline. Intrinsic in the selection process lies not only *how* to convert the data to the desired format, but also *which* data should be included, and which should be omitted.

The mathematical object most commonly used for structured data in visualization is the *data table* [22], which consists of a set of relations (expressed as tuples) and metadata describing the relations. Each tuple records a specific *case* in the data set (i.e. a person, a movie, a time-stamped measurement, etc), and each entry in a tuple represents a *variable* (i.e. the person's name, the movie title, the time stamp, etc). The data type of a variable will have bearing on its visual encoding further on in the pipeline. There are three basic variable types:

- *quantitative* – supports arithmetic (e.g. a temperature value);
- *ordinal* – obeys an ordering relation (e.g. the days of the week); and
- *nominal* – supports only equality and non-equality (e.g. movie titles).

Encoding

The *encoding* transformation accepts structured data (i.e. a mathematical object) and generates a *visual object* representing it. The actual mapping between mathematical and visual object (typically called *visual mapping*) is application-dependent. For visualization or scientific visualization, it may be a straightforward transformation to spatial form, whereas for information visualization, the choice of representation depends on which aspects of the data the designer wants to highlight to the user; even then, much of the details of the visual representation are left in the hands of the designer. Herein lies the great challenge of information visualization (as opposed to the other visualization disciplines): to design a suitable visual representation that clearly and concisely captures the data we want to display.

We formulate these two criteria in the concepts of *expressiveness* and *effectiveness* [97]. An encoding is said to be *expressive* if all and only the data in the mathematical object can be represented in the visual object. Furthermore, the encoding is *effective* if it fully exploits the capabilities of the output medium and the human visual system (effectiveness is often used for comparing two different visualizations).

A visual object consists of a *spatial substrate*, *marks*, and the *graphical properties* of the latter [12, 21, 22, 97]. Marks can be combined using a simple composition algebra that includes operations like connection and enclosure. A simple visualization includes a number of marks (points, lines, or volumes), their retinal properties (color, texture, and size), and their positions on the spatial substrate (obeying the orientation and placement of the axes on the substrate). In the example of a scatterplot, the substrate is composed of two orthogonally placed axes (one for each of the variables being expressed), creating a 2D Cartesian space, and a number of point marks representing the cases.

Figure 1.2 summarizes the language of graphical encoding. Using this language, we can for instance express a tree diagram as a composition of point marks connected by line marks on a 2D spatial substrate. Having defined this language, the step to automating the graphical encoding process is not far; see for example Mackinlay's work in this area [97].

Presentation

Even if we have now chosen a visual representation of our data and encoded the data into a visual object, we still need to *present* the visual object to the user, to

- **Marks:** points, lines, areas, volumes, etc.
- **Properties:**

  Positional: 1D, 2D, 3D.

  Temporal: animation.

  Retinal: color, shape, size, saturation, texture, orientation.
- **Compositions:** connection, enclosure.

Figure 1.2: A simple graphical language.

create one or several *views* of the data. Views in visualization are almost always interactive, allowing us to exploit the time parameter to extract more information out of the visualization than would be possible from a static diagram. Often it is not even feasible to view the entire visual object due to its complexity or size; thus, we have to employ various *view transformations* to allow us to see relevant details of the data as well as getting an overview of the whole or parts of the data set.

According to Card et al. [22], there are three common view transformations:

- *Location Probes:* Show details (often in a separate window) of the data set at specific points in the visual structure chosen by the user (details-on-demand).
- *Viewpoint Controls:* Provide controls to allow the user to zoom in and pan around a detailed view of the data set (for example, scrolling in a text document). Also provide an overview of the data set to prevent the user from getting lost [123].
- *Distortion:* Distort the spatial substrate so that the detail view and the overview are combined in the same space, creating a so-called focus+context view [59].

Interaction

Finally, the main distinguishing feature of visualization over static diagrams is the existence of human *interaction* feedback into the visualization pipeline. For every transformation in the model, there is a conceivable human interaction to allow the viewer to manipulate parameters in the visualization: guiding the selection process, mapping variables to visual objects in the encoding process, or controlling the views in the presentation.

1.1.2 Visualization Techniques

Based on the visualization model described above, we can now define the concept of a *visualization technique* as consisting of a visual object structure, an encoding function that accepts structured data and generates a visual object, a variable set of views, and a number of interaction techniques for manipulating the encoding function as well as the presentation (the views).

For information visualization, the technique may also have a metaphor associated with it, primarily for the benefit of the users; for instance, modern user environments often use the desktop metaphor (with a workspace, trashcan, folders, files, etc). The purpose of this is to give users a familiar handhold in an otherwise alien environment, and to provide the user with some free knowledge about it (for instance, that folders can be opened and that files can be deleted by putting them in the trashcan).

1.2 Information Visualization

Our definition of *information visualization* is a specialization of our previous definition of visualization, and amounts essentially to the following [22]:

> **Information visualization:** The use of computer-supported, interactive visual representations of abstract data to amplify cognition.

As noted earlier, the distinguishing feature of information visualization is that the data we are visualizing no longer has a natural mapping to a graphical form. Instead, the visualization designer has to invent this visual mapping and create a graphical representation that somehow captures the underlying data in such a way that the user's understanding is improved.

Although many of the practices associated with information visualization have been in use for a long time, the field itself is generally recognized to have been defined as late as 1993 by Robertson et al. [119]. Classic early examples of information visualization include Charles Minard's (1781-1870) famous graph of Napoleon's failed campaign against Russia in 1812, William Playfair's (1759-1823) invention of the line plot, bar chart, and pie chart in 1786 to show the balance of trade between countries [112], and Florence Nightingale's (1820-1910) striking rose-like "Coxcomb" visualization from 1858 showing that far more deaths in the Crimean War were attributable to non-battle causes than battle-related causes. These and more examples of famous visualizations can be found in Edward R. Tufte's excellent book [137].

1.3 Contributions

The work in this thesis deals with the general research area of visualization, but has taken two separate and quite different turns:

- **3D Occlusion Management.** General techniques for reducing the impact of the occlusion effect on task solving in 3D environments.
- **Causality Visualization.** Techniques and applications of information visualization for complex causal relations.

In the following sections, we will briefly review each of these two parts in more detail and outline the major contributions in each included paper.

1.3.1 3D Occlusion Management

The first part of this thesis identifies and attacks the problem of *3D occlusion management*, i.e. how to efficiently and intelligently avoid (or at least reduce) performance degradation of visual perception tasks in 3D environments due to inter-object occlusion. That nearby objects hide distant objects is a fact of life in three-dimensional worlds, but techniques in this area directly or indirectly aim to counterbalance the impact of this phenomenon on the work of the human user in the environment. The major contributions of this work can be summarized as below:

- the identification and formalization of the occlusion management problem in 3D environments;
- a taxonomy of 3D occlusion management techniques, both new and existing; and
- four new and orthogonal techniques for 3D occlusion management, all of them empirically evaluated using formal user studies:
  - view projection animation;
  - interactive 3D space distortion;
  - image-space dynamic transparency; and
  - 3D navigation guidance.

Chapter 2 sets the stage by defining the occlusion problem and the general occlusion management approach taken in this thesis. A considerable amount of related work exists that directly or indirectly attacks the occlusion problem; in **Chapter 3**, we systematize 25 of these techniques into a taxonomy of 3D occlusion management techniques based on six different parameters [51]. These techniques then form the basis for deriving five elementary design patterns for occlusion management, four of which we employ in subsequent chapters in this thesis.

The view projection animation technique [50] presented in **Chapter 4** is an example of a projection distorter, smoothly manipulating the camera transformation matrix to create a "dolly-and-zoom" effect. The motivation behind this technique is to provide a spring-loaded parallel projection glance [111] for situations when nearby objects take an inordinate amount of screen space; parallel projection mode discards the depth coordinate of objects, assigning them screen space proportional only to their geometrical size. Our user study shows a significant improvement for correctness in visual perception tasks, but at the cost of a significant additional time investment.

The 3D space distortion technique called the BalloonProbe [44] presented in **Chapter 5** is an example of the interactive exploder design pattern, and acts somewhat like a 3D magnifying glass. Designed for use primarily in immersive virtual environments, the interaction technique builds on an inflatable virtual probe connected to the user's input device. Inflating the probe (typically as a balloon-like 3D sphere) will displace all objects it collides with to the surface of the probe; deflating it will return them to their original positions. In **Chapter 6** we empirically compare [52] two kinds of BalloonProbes to two 3D fisheye [59] implementations as well as standard 3D navigation in a CAVE device [35]; results again indicate a clear trade-off between speed and accuracy for visual perception tasks in 3D environments. Nevertheless, spherical BalloonProbe turned out to be the most accurate interaction technique.

Virtual X-Ray techniques promote target discovery by changing the transparency of occluding surfaces; in **Chapter 7** we present our *dynamic transparency* method for achieving this as well as an image-space real-time rendering algorithm that implements it [6]. The algorithm is based on modern programmable graphics hardware and uses several rendering passes to achieve an effect similar to superhero "X-Ray vision", creating a transparency gradient outline around all occluded targets. To evaluate the viability of the dynamic transparency approach as a whole, we also performed a user study involving four tasks in two different kinds of 3D environments (abstract and realistic). The results clearly show that the new method is superior in terms of both efficiency as well as correctness.

Finally, the last chapter in this part of the thesis, **Chapter 8**, deals with navigation guidance for 3D exploration [49]. This method is an example of the tour planner design pattern, where the approach is based on exchanging image-space or object-space methods for temporal space, i.e. sequentially showing the targets using a carefully constructed tour through the 3D environment. Our algorithm consists of two distinct phases: an off-line precomputation step that actually builds the grand tour, and an on-line interaction technique that we call spring-zooming for exploring the environment using the tour. Again, we evaluated this method using an empirical user experiment, studying not only whether guidance helps, but also whether having too much guidance might negatively affect the user's learning of the 3D environment. While no evidence was found of the latter, the evaluation showed a significant improvement in both world recall and search performance for subjects using the new spring-zooming technique.

1.3.2 Causality Visualization

The second part of this thesis deals with *causality visualization*, or the graphical representation of causal relations to amplify cognition. This is a classical information visualization problem, with potentially huge and complex datasets of time series data. The work consists of two different visualization techniques developed for this problem based on the concept of growing 2D geometry [47] for representing time. The major contributions of this work are the following:

- the Growing Squares technique for animated 2D visualization of causal relations;
- the Growing Polygons technique for animated 2D visualization of causal relations; and
- the CiteWiz system for scientific citation network visualization using the Growing Polygons technique (as well as two other).

The Growing Squares [46] technique, presented in **Chapter 11**, was the first approach and uses a combination of color and animation to represent the processes in a concurrent system (such as a computer network) as color-coded squares on a drawing canvas. The time parameter is mapped to the geometrical size of each square, making them grow as the visualization is animated. Messages from one process to another carry the color from the source to the sink process, showing influence patterns for each process similar to age rings in a tree. We conducted a user study comparing the new technique to standard time-space causality diagrams. The results showed a significant improvement for Growing Squares only for small datasets, not for the general case, indicating a scalability weakness.

As a consequence, we present the Growing Polygons [45] technique in **Chapter 12** that addresses some of the problems of the previous technique, exchanging the squares for n-sided process polygons scattered around the perimeter of a large layout polygon. By devoting a specific portion of each polygon to a specific process, the visualization becomes significantly easier to read and understand. Accordingly, results from a user study similar to the one conducted before confirmed this fact—subjects were now significantly faster at solving causality tasks using our technique compared to time-space diagrams for all dataset sizes.

Causality is a phenomenon with a wide range of applications; in **Chapter 13** we show a case study employing the Growing Polygons technique for the visualization of scientific citation networks in the CiteWiz [48] system. The tool uses an XML-based citation database to present influence views of papers and authors using the causality visualization technique. It also has two additional presentation modes designed to help the user get an overview of a citation network: a glyph-based static timeline visualization, and an interactive concept map with a dynamic force-directed layout scheme. The paper also presents a taxonomy for citation database usage based on roles, goals, and tasks. A user study indicates a significant efficiency improvement for subjects using the CiteWiz tool for complex citation tasks compared to standard web-based database interfaces.

Part I

3D Occlusion Management

Chapter 2

Introduction

Human beings employ all manners of visual cues and hints in order to correctly perceive and understand the three-dimensional world surrounding us. Of course, these visual cues can also work against us, fooling our perception into believing things about our environment that are simply not true. In some cases, this is done intentionally through various forms of optical illusions that exploit special characteristics of our minds. A more subtle point, however, is that we can instead choose to directly weaken certain of these visual cues in order to help the human to perceive and understand *more* of her surroundings. In some cases, this selective weakening of visual cues, primarily occlusion, size, and shape, can lead to dramatically increased performance when solving specific tasks in a 3D environment. While this may be difficult to achieve in the real world, it is a perfectly viable approach in a virtual 3D world being visualized on a computer.

In this chapter, we describe the *occlusion* effect in 3D environments, i.e. the fact that nearby objects hide more remote objects. We then go on to introduce the *occlusion management problem* for intelligently controlling the environment and its views in such a way that the impact of occlusion on the performance and correctness of a user solving visual perception tasks in the environment is reduced or, in some cases, even eliminated.

That occlusion occurs in three-dimensional environments is a fact of life, one that we humans are well-familiar with from literally thousands of years of evolution. However, when it occurs in a computer-generated 3D world as opposed to our real one, we can actually do something about it. This is the leap of faith that this thesis takes—to probe the borders of our usual perception of visual cues in order to aid us in understanding our surroundings. In the following chapters, we shall see how.

2.1 Problem Space

The occlusion problem space in 3D environments is defined by the intrinsic *properties* of the environment, their interaction with *human cognition*, the *visual tasks* involved, and the ensuing effects caused by the occlusion. The environment and its geometrical properties interact with human vision, causing occlusion of objects and leading to loss of correctness and productivity.

2.2 Model

We represent the 3D world U by a Cartesian space $(x, y, z) \in \mathbb{R}^3$. Objects in the set O are volumes within U (i.e. subsets of U) represented by boundary surfaces (typically triangles). The user's viewpoint $v = (M, P)$ is represented by a view matrix M that includes the position and orientation of the user, as well as a projection matrix P that includes view parameters such as viewport dimensions, focal length, far and near clipping plane, etc.

A line segment r is *blocked* by an object o if it intersects any part of o. An object o is said to be *occluded* from a viewpoint v if there exists no line segment r between v and o such that r is not blocked. Analogously, an object o is said to be *visible* from a viewpoint v if there exists a line segment r between v and o such that r is not blocked. An object o is said to be *partially occluded* from viewpoint v if o is visible, but there exists a line segment r between v and o such that r is blocked.

An object can be flagged either as a *target*, an information-carrying entity, or a *distractor*, an object with no intrinsic information value. Importance flags can be dynamically changed. Occluded distractors pose no threat to any analysis tasks performed in the environment, whereas partially or fully occluded targets do, potentially causing decreased performance and correctness.

Figure 2.1 shows a diagram of this basic model formulation. Here we see three objects A, B, and C, the first of which is a distractor and the other two targets. The shaded area represents areas invisible to the user from the current view. It is easy to see in this diagram that A is fully visible (but is a distractor), B is partially occluded, and C is occluded.

A set of viewpoints V is said to be *complete* if there exists no object that is occluded in all of the viewpoints v_i. For instance, from the figure it is clear that the set $V = \{v_0, v_1\}$ is complete for the simple environment given in the example.

It is possible to introduce a temporal dimension to this model and discuss concepts like transient occlusion and invariant occlusion. We will ignore this aspect in this thesis, however, and consider only temporally invariant situations. Some of the solutions we develop will still be applicable to dynamic situations.

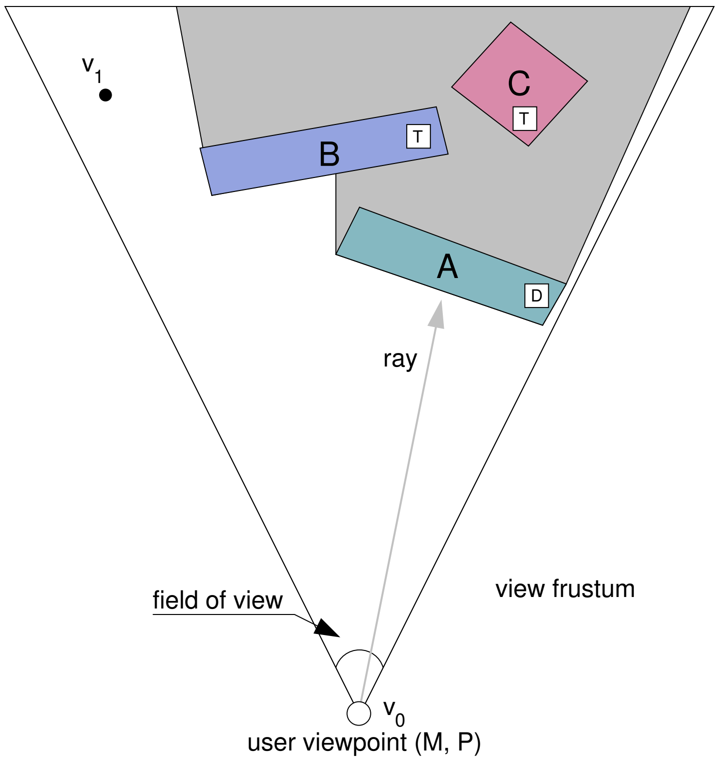


Figure 2.1: Basic model for 3D occlusion.

2.3 Visual Tasks

The occlusion problem typically occurs in the following three *visual tasks*:

- *object discovery* – finding all targets $t \in O$ in the environment;
- *object access* – retrieving graphically encoded information associated with each target; and
- *spatial relation* – relating the spatial location and orientation of a target with its context.

Other visual tasks that are of relevance include object *creation*, *deletion* and *modification*; in this treatment, however, we consider these to be special cases of discovery and access with regards to inter-object occlusion, and consisting of the same subtasks as these three basic visual tasks.

2.4 Analysis

We can observe that all visual tasks are severely hampered by the existence of fully occluded objects. More specifically, for the purposes of object discovery, a fully occluded object will be impossible to discover without the use of some occlusion management strategy, and identifying whether the object is a target never becomes an issue. Analogously for object access, the visual search will fail, and so will the perception of the object's visual properties. As a result, both tasks will affect the efficiency and correctness of users solving tasks using a visualization, but clearly, threats to object discovery are the most serious: if the user is unaware of the existence of an object, she will have no motivation to look for it and access never becomes an issue.

Partial occlusion, on the other hand, has a different effect on these tasks. For object discovery, users may have difficulties distinguishing object identity if too large a portion of the object is occluded. In this situation, the user may either miss the object entirely, count the same object multiple times, or believe different objects are part of the same object. Object access, on the other hand, will succeed in the visual search, although the perception of the object may still fail due to important parts of it being occluded.

Spatial relation, necessary for many complex interactions and visualizations, requires overview of the whole world, and is thus severely affected by both partially and fully occluded objects.

2.5 Environment Properties

The geometrical properties of the visualization environment are of special interest in this framework because they allow us to characterize the visualization and determine the nature of the occlusion problems that may arise. These properties can also be used to decide which occlusion management strategies are applicable for a specific situation.

In this treatment, we identify three main geometrical properties of the environment that interact to cause inter-object occlusion and influence the three basic visual tasks associated with the environment:

- *object interaction* – spatial interaction of objects in the environment;
- *object density* – amount of objects in the environment with regard to its size; and
- *object complexity* – detail level of individual objects in the environment.

Obviously, these are high-level properties that only generally describe an environment without going into detail on its actual content. Nevertheless, in the following sections we shall see how these property dimensions can serve as powerful reasoning tools for describing a 3D environment and selecting a suitable solution strategy for it.

2.5.1 Object Interaction

The object interaction property dimension describes how the individual objects in the environment interact spatially with each other, i.e. whether they touch, intersect or merely reside close to each other. There are five ordinal levels to this parameter (see Figure 2.2 for a visual overview):

- *none* – no spatial interaction between objects (realistically only applicable for singleton objects);
- *proximity* – objects are placed in such close proximity (without intersecting) that they occlude each other from some viewpoint;
- *intersection* – objects intersect in 3D space (without one fully containing another) such that they occlude each other;
- *enclosement* – one or several objects combine to fully enclose objects (without containing them) such that they are occluded from any viewpoint external to the enclosing objects; and
- *containment* – objects are fully contained in other objects such that they are occluded from any viewpoint.

Examples of these interaction levels exist in all kinds of 3D visualizations: proximity for nodes in 3D node-link diagrams, intersection for visualization of constructive solid geometry (CSG), enclosement for 3D objects placed inside larger objects (i.e. the walls of a virtual house), containment for 3D medical CAT scan data, etc.

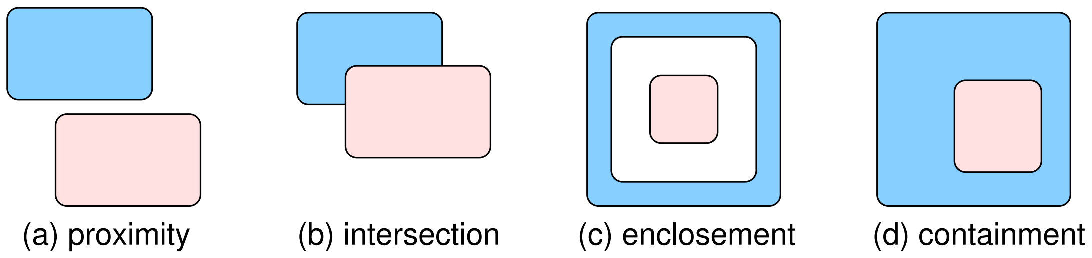


Figure 2.2: Object interactions that may cause occlusion in 3D environments.

2.5.2 Object Density

The object density is a measure of the number of objects inhabiting the 3D environment; it follows naturally that the more objects per volume unit we are dealing with, the greater the chance and impact of occlusion will be. For environments containing a singleton object, naturally only self-occlusion can occur.

2.5.3 Object Complexity

The third geometrical property with an impact on the occlusion characteristics of an environment is the complexity of the objects in the environment. With complexity, we refer to the detail level of the 3D objects, i.e. typically the number of triangles (or other 3D primitives, such as quads, lines, and points) that make up the object, but we also include attributes such as color, material, and texture in this parameter. It follows that the more complex an object is, the more information it can potentially encode, and the larger the impact occlusion has on identification and perception of the object.

For simplicity, we can often reduce object complexity by splitting objects into smaller (preferably convex) subobjects. Note that this will often result in an increased object interaction and density index. The same mechanism can be used to handle self-occlusion, i.e. when an object occludes parts of itself.

2.6 Visual Cues

It is clear that we cannot go around changing or weakening the visual cues in a 3D world without stopping to consider the implications of our actions. Visual cues serve an important role in helping us decipher our surroundings, so modifying them might have the opposite effect we intended. In this section, we will discuss these visual cues in more depth;[1] the information here is primarily based on material found in [62, 65, 80, 109].

The term "visual cue", though imprecise, has become the standard way of discussing visual space perception, the reason being the complexity and heterogeneity of describing the concepts involved. The definition we will prefer here is the following: a *cue* is a complex interrelationship (or several relationships) between aspects such as external objects, impinging light energy, physiological excitation, and assumptions. The descriptive and informal nature of the term is sufficient for our treatment, however.

Figure 2.3 gives an overview of the visual cues humans use to disambiguate depth (the exact number and naming of these is not standardized; the list found here is a combination of the work of several authors). As can be seen from the table, we classify these cues according to two primary dimensions:[2]

- whether the visual cue relies on one or both eyes (monocular versus binocular); and
- whether the cue is psychological (perceptive) or physiological (proprioceptive) in nature.

[1]No pun intended.

[2]As noted by Ittelson [80], however, the classification of these concepts may be more or less fruitless due to the difficulty of separating the processes involved in each category.

	perceptive	proprioceptive
monocular	occlusion, relative size, familiar size, relative height, texture gradient, shadow, linear perspective, aerial perspective, brightness, motion parallax	accommodation
binocular	stereopsis (static parallax)	vergence

Figure 2.3: Overview of visual depth cues.

We will discuss these categories in more depth in the following text. Note that a detailed description of the physiology of the human eye is beyond the scope of this thesis; nevertheless, this knowledge may come in useful when reading.

2.6.1 Proprioceptive Cues

Proprioceptive cues, also called *oculomotor cues*, come from physiological feedback from the muscles of the actual eye itself. Therefore, the effect is most useful for short distances—for longer distances, the feedback is less tangible, and thus insufficient for depth estimation. Since human beings do not spend so much time looking at close objects, proprioceptive cues are not very important for our visual systems, but they are vital for certain animals (especially those with lateral eye placement, such as rabbits and chameleons).

Accommodation

Accommodation is a monocular proprioceptive depth cue derived from feedback from the eye muscles responsible for focusing the eye's crystalline lens on an object (the *ciliary muscles*). The strain on these muscles to focus on the object gives an indication to the brain on the distance of the object; the closer the object, the more the ciliary muscles must contract to change the curvature of the crystalline lens to bring the object properly into focus. At longer range (more than two meters), the differences in muscle contraction are too small for the feedback to be of any use.

For humans, the accommodation cue is insignificant and typically only employed in conjunction with other visual cues.

Vergence

The binocular proprioceptive depth cue is called *vergence*, and is produced from feedback from the movement muscles of both eyes focusing on an object. More specifically, as the brain focuses on an object, the eyes are controlled so that they intersect on the object; the angle at which they cross is the vergence. For close objects, the angle is large, whereas for more distant objects, the view direction of both eyes becomes almost parallel. This again means that vergence cues are less useful for longer distances.

The concept of vergence can also include a dynamic property; a focused object moving away from the viewer causes *divergence*, whereas moving towards the viewer causes *convergence*.

2.6.2 Perceptive Cues

The other main category of visual cues is *perceptive*, or psychological, cues. As opposed to proprioceptive cues, these arise from the subconscious processing performed by the human visual system to interpret and "stitch together" the information collected from the eyes.

There is only one binocular perceptive cue, *stereopsis*; the others are all monocular in nature, i.e. can be perceived using only one eye. Due to this fact, monocular cues have traditionally been employed as important ways of conveying depth and realism for artists and painters, who are more or less constrained to flat, two-dimensional canvases. Therefore, this class of depth cues is often referred to as *pictorial cues* [65] (with the exception of motion parallax, which requires a dynamic medium impossible for a painting). For interactive and immersive display systems such as CAVEs [35] and HMDs [133], even binocular cues are accessible and can be employed to great effect.

Figure 2.4 shows the nine pictorial cues from the table above; we will describe them (as well as the stereopsis effect) in greater depth below.

Stereopsis

Stereopsis, also called *static* or *binocular parallax*, arises from the *binocular disparity* of our visual system, i.e. the fact that the two pictures seen from our two eyes are slightly different (simply due to them being placed apart). The human visual system is capable of taking these two pictures and combining them into a single, integrated picture, allowing for depth perception at close and medium range.

Stereopsis is closely related to motion parallax covered below, with the exception that static parallax does not require motion for depth perception, whereas motion parallax instead performs the same "stitching" process in temporal space by combining two consecutive images into a single three-dimensional one.

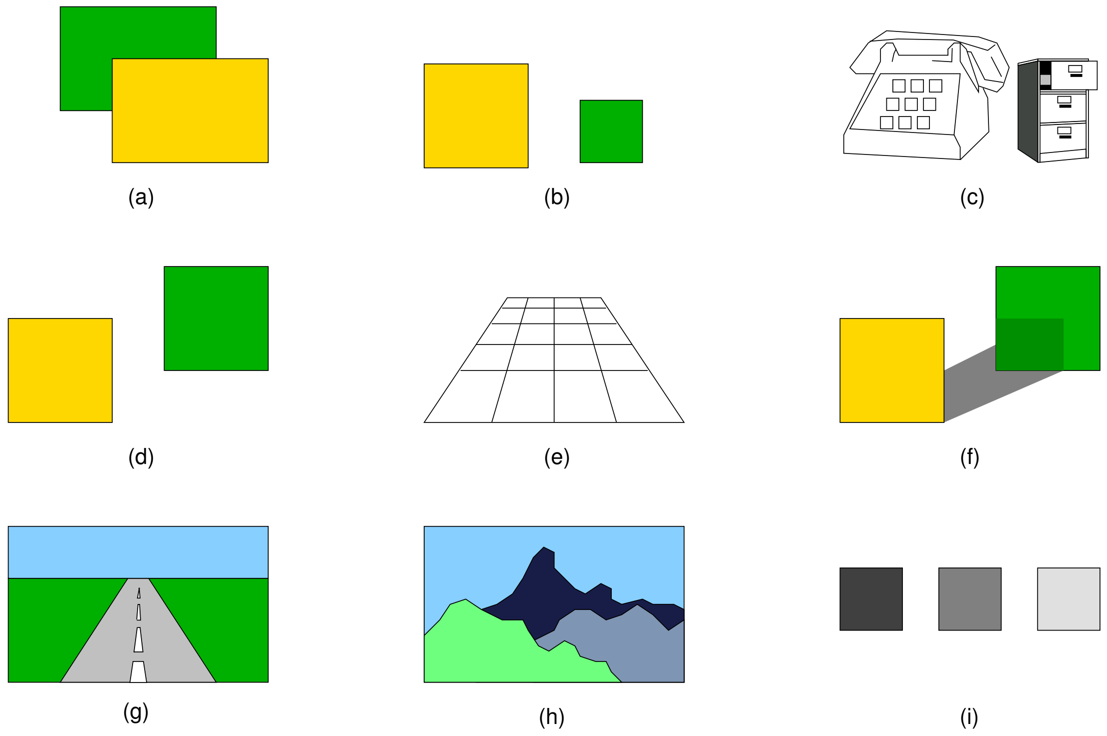


Figure 2.4: Pictorial depth cues.

Occlusion

The *occlusion* cue, also known as *overlapping*, *interposition*, *superposition*, or *overlay*, is of special relevance to this thesis. This occurs when two objects of different distance come into conflict of each other so that the closer object partially obscures the distant object. In other words, when an object partially hides another object, the hidden object is perceived as being further away. See Figure 2.4(a) for an example: the yellow box is perceived as being closer to the viewer than the occluded green box.

Note that occlusion is merely a relative depth measurement; it only gives a distance relationship between two objects, not their absolute distance from the viewer, nor the actual distance between them.

Relative Size and Familiar Size

The retinal size of an object viewed by a human, its *relative size* cue, gives an indication of its distance in comparison to another object. In Figure 2.4(b) we see that the larger yellow square, all other things being equal, is perceived as being closer than the smaller green square. The automatic assumption is that the two objects are of similar size, but that perspective foreshortening is making the green square look smaller due to its longer distance from the viewer.

In Figure 2.4(c), however, the objects involved have a familiar meaning, and this assumption can no longer be made. The *familiar size* depth cue allows previous knowledge of the involved objects to help us disambiguate depth. In the example, it is clear that the telephone is closer than the file cabinet simply because we know empirically that a file cabinet is normally much larger than a telephone.

In fact, unlike relative size, familiar size is an absolute depth cue. Given that we know the typical size of an object and the retinal size it currently takes on our field of view, we can make an estimation of its distance. Of course, this fact can easily be utilized to fool the perception system by scaling up or down a familiar object to an unusual size.

Relative Height

Relative height, also known as *position in the field*, is a visual cue related to the relative positioning of perceived objects in the view. The closer an object is to the horizon, the further away it is perceived to be. For things placed on the ground, this translates to objects positioned higher in the picture being seen as more distant than ones positioned lower (the green and yellow squares in Figure 2.4(d), respectively). The relation is reversed for things located in the air, i.e. above the horizon.

Texture Gradient

Related to the relative size depth cue, *texture gradient* is a visual cue in its own right that is exhibited in the diminishing size of subobjects and smoothness of a surface. The grid in Figure 2.4(e) is an example of a pattern that seems to stretch away into the distance due to the perspective foreshortening effect on the individual grid squares.

Shadows and Shades

Prior experience tells us that given two objects, the one closest to a light source may cast a *shadow* on any object partially or fully behind the object (in relation to the light source). See Figure 2.4(f) for an example; the yellow square is again perceived to be closer due to it casting a shadow on the green square. Experience furthermore tells us that most light comes from above; therefore, the *shading* of an object on itself also gives indications of its depth.

Linear Perspective

The visual cue of *linear perspective* arises from rotating parallel lines toward each other, creating the illusion of depth and perspective. Figure 2.4(g) shows how two angled lines give the effect of a straight road receding into the distance. Invented by 15th Century Renaissance artists, this depth cue is closely related (although different!) to both relative size and texture gradient.

Aerial Perspective

Aerial, or *atmospheric*, *perspective* appears when looking at objects in the far distance, causing the interference of air, dust, and humidity to make the object appear less sharp and a little bluish. The amount of haziness and bluishness is dependent on the actual distance; the further away, the more atmosphere lies between the object and the viewer, making the picture less clear. Figure 2.4(h) illustrates this with three different mountain ridges at different distances with increasing amounts of blue tint.

Brightness

Related to aerial perspective, Figure 2.4(i) shows an example of the *brightness* visual cue causing a lighter object (the light gray square on the right) being perceived as being closer than the darker objects.

Motion Parallax

Finally, we have the concept of *motion parallax*, which is a dynamic visual cue arising during the observation of the movement of two different objects at different distances. This typically occurs when the viewer is moving. Due to the peculiarities of perspective foreshortening, the closer object will appear to move faster than the more distant one. The effect is particularly strong when looking out the side window of a moving train or car.

Other dynamic effects occurring in the context of motion parallax are *deletion* and *acretion*, the dynamic processes of a distant object being occluded or revealed, respectively, as two objects at different distances move in relation to each other.

2.6.3 Discussion

The techniques in this thesis mainly target the occlusion perceptive cue from the treatment above. However, this is still important background material for understanding the impact of our methods, if only to ensure that all other visual cues are retained even if we relax or eliminate the occlusion cue. This may be of particular importance for the image-space dynamic transparency technique described in Chapter 7—if we were to blindly uncover hidden objects with no regard to these visual cues, we could easily get the phenomenon of *reverse occlusion*, when instead distant objects all of a sudden occlude nearby ones.

In addition, knowledge of visual cues is vital for generalizing this work to the more ambitious approach of *augmented perception*, i.e. general techniques for aiding the human in all perceptual tasks in a 3D world. It is clear that all of the above visual cues may be fair game for these kinds of techniques, not just occlusion.

2.7 Solution Space

Given the above description of the occlusion effect and its parameters, we can define the corresponding *occlusion management problem* as the following:

> **Occlusion management:** Manipulation of the 3D environment and its views with the purpose of reducing or eliminating the impact of inter-object occlusion on representative visual perception tasks.

This problem is the subject of the following chapter.

Chapter 3

Taxonomy of 3D Occlusion Management Techniques[1]

Niklas Elmqvist[2], Philippas Tsigas[2]

Abstract

While an important factor in depth perception, the occlusion effect in 3D environments also has a detrimental impact on tasks involving discovery, access, and spatial relation of objects in a 3D visualization. A number of interactive techniques have been developed in recent years to directly or indirectly deal with this problem using a wide range of different approaches. In this paper, we build on previous work on mapping out the problem space of 3D occlusion by defining a taxonomy of the design space of occlusion management techniques in an effort to formalize a common terminology and theoretical framework for this class of interactions. We classify a total of 25 different techniques for occlusion management using our taxonomy and then go on to analyze the results, deriving a set of five orthogonal design patterns for effective reduction of 3D occlusion. We also discuss the "gaps" in the design space, areas of the taxonomy not yet populated with existing techniques, and use these to suggest future research directions into occlusion management.

Keywords: occlusion management, occlusion reduction, taxonomy, design patterns, visual cues, depth perception

[1]This is an extended version of a paper that will appear in *Proceedings of the IEEE Conference on Virtual Reality 2007*, Charlotte, North Carolina, USA.

[2]Department of Computer Science, Chalmers University of Technology, 412 96 Göteborg, Sweden. E-mail: {*elm,tsigas*}*@cs.chalmers.se*

3.1 Introduction

In this paper, we explore the design space of interaction techniques that perform *occlusion management* by modifying certain depth cues in order to increase the spatial awareness of the human user and to facilitate special tasks, such as navigating, searching, or understanding the 3D world. More specifically, we present a taxonomy consisting of a small set of dimensions describing important characteristics of these techniques, focusing on the purpose, strength, view paradigm, depth cues, interaction model, and preserved invariances of each technique. We then go on to classify 25 different methods that have been described previously in the literature into the taxonomy. These classifications form a body of data that we can analyze for trends and the existence of clusters; this analysis yields in turn five orthogonal *design patterns* that characterize current work in the field. The patterns are multiple viewports, virtual X-ray tools, tour planners, interactive exploders, and projection distorters, and we describe the typical uses and characteristics of each pattern. More importantly, the pattern identification process also serves to pinpoint the "gaps" in the taxonomy, i.e. as-of-yet undeveloped techniques that could potentially fulfill a useful role in future research.

The purpose of this taxonomy is manifold: (i) to provide a common theoretical framework and vocabulary for occlusion management techniques, giving researchers and practitioners alike a common ground for discussion; (ii) to facilitate qualitative comparison, evaluation and maybe even benchmarking of different methods for occlusion management; (iii) to suggest a small number of archetypes of design suitable as starting points for implementations and prototypes; and (iv) to inform future directions of research within occlusion management and human perception of 3D space.

3.2 Previous Taxonomies

No previous taxonomy exists in the literature on the class of occlusion management interaction techniques. More general taxonomies on 3D interaction tend to describe low-level mechanics of manipulative tasks in a morphological fashion, whereas our focus is more on high-level aspects of perceptual tasks related to spatial understanding of the 3D environment. For example, Bowman and Hodges [16] present a general formal framework for 3D interaction in immersive virtual environments (IVEs) based around three tasks: motion control, selection, and manipulation. Bier et al. give a taxonomy of see-through tools [13] for a class of double-handed interaction techniques using transparent sheets called toolglasses that served as inspiration for this taxonomy. Bowman et al. [18] present a descriptive view of the design space of information display as well as interaction for information visualization within virtual environments.

Although unrelated to the occlusion management area defined here, Pousman and Stasko's taxonomy of ambient visualization [114] inspired the method employed in this paper for deriving the design patterns from the classification data.

We use our taxonomy as a tool for classifying existing techniques and thus validating its generality, but also as a design space. This allows us to identify holes in the taxonomy, akin to [20].

3.3 Design Space

We characterize the design space of occlusion management techniques using the following primary dimensions:

- **Primary Purpose.** Visual task that the technique is primarily targeting. [discovery, access, relation]
- **Disambiguation Strength.** Maximum object interaction that the technique can handle. [proximity, intersection, enclosement, containment]
- **Depth Cues.** Strength of depth disambiguation cues for the technique. [low, somewhat low, medium, somewhat high, high]
- **View Paradigm.** View method used for the technique, i.e. the arrangement and layout of the visual substrate. [single view, twin separate views, twin integrated views, multiple separate views, multiple integrated views]
- **Interaction Model.** Operational model of user interaction for the technique. [passive, hybrid, active]
- **Target Invariances.** Degree of target invariances preserved using the technique. [0–3 aspects: location, geometry, appearance]

These six dimensions have been identified to be orthogonal, objective, and capture the full expressivity of the design space of these kinds of techniques. In the following sections, we will describe the dimensions in greater detail.

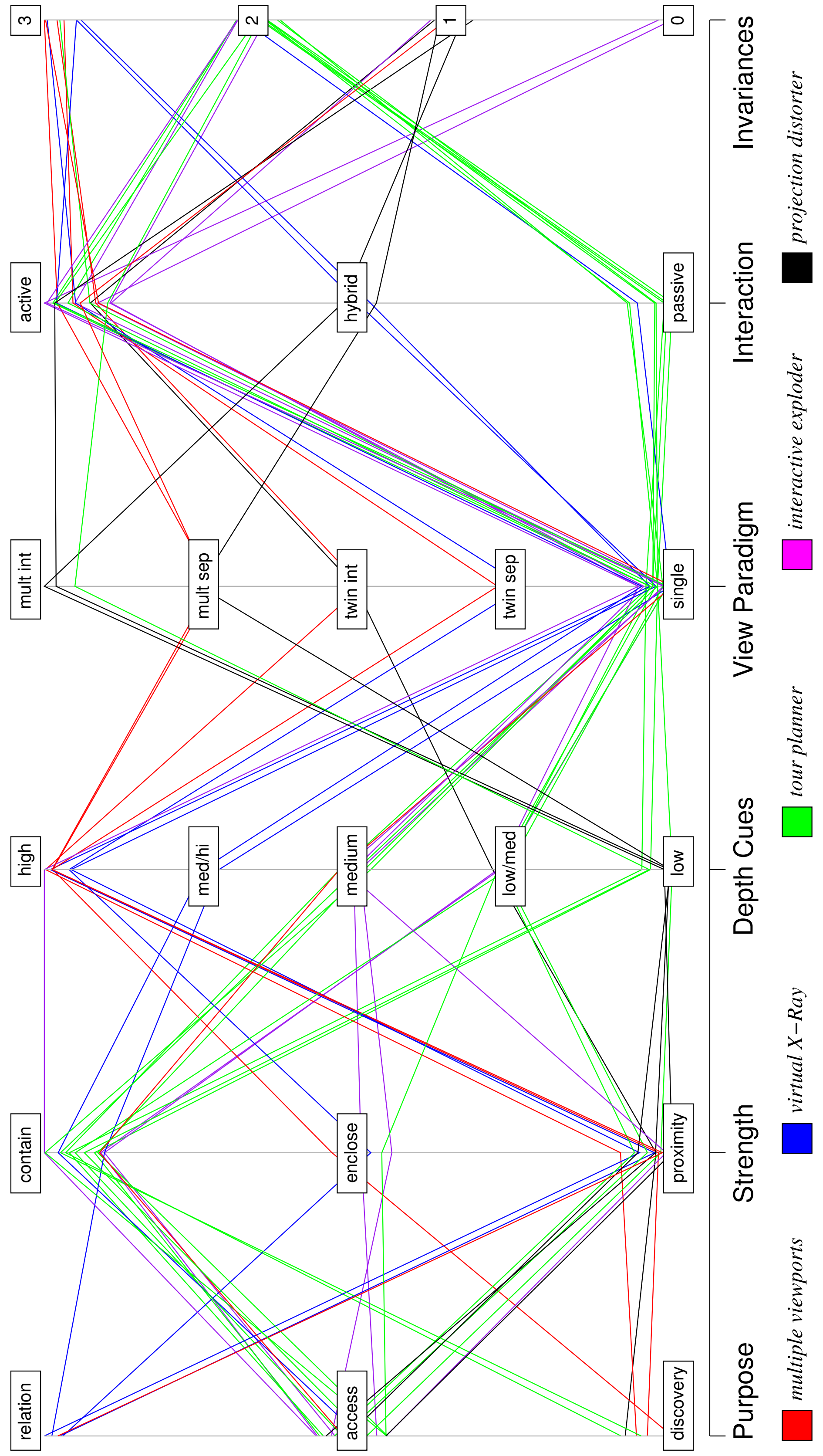


Figure 3.1: Classification of 25 different occlusion management techniques using the taxonomy (points have been jittered to show distribution).

3.3.1 Primary Purpose

The purpose of an occlusion management technique describes which particular visual task in the problem space that the technique is primarily targeting (see Section 2.1 for the visual tasks). In other words, this dimension can assume any of the values discovery, access, or spatial relation.

More specifically, an interaction technique designed mainly for discovery focuses on making the user aware of the existence of partially or completely occluded targets, not necessarily making retrieval or relation of information from the objects easier.

A technique designed for access, on the other hand, aims not only to make users aware of an occluded object, but also to allow the user to retrieve the information encoded in the object.

Finally, a technique supporting spatial relation is designed to make not only the object itself but also its surrounding context visible and understandable to the user. This means that it is not possible to simply get rid of the neighboring objects in the interest of seeing the target, since these may carry important information needed to understand the scene (such as the connectivity of a node-link diagram).

Note that a technique may have more than one purpose; this taxonomy dimension captures the **primary** purpose of the technique.

Domain: *discovery, access, spatial relation* (nominal)

Characteristic Techniques:

- *discovery:* image-space dynamic transparency [6]
- *access:* interactive cut-away and break-away views [43], 3D explosion probe [130]
- *spatial relation:* tumbler [116], way-finder [4]

3.3.2 Disambiguation Strength

Disambiguation strength refers directly to the maximum degree of object interaction that the technique can handle and still fulfill its primary purpose. In other words, this is a measure of how complex object interactions the technique can manage using the terminology from the problem space (see Section 2.5.1). Note that this metric is unrelated to object density, but that very high object density can confound the situation.

The strength of a technique is an ordinal dimension, and it is generally perceived better for a technique to be able to handle high object interaction. On the other hand, strength is related to other factors of the design space, leading to a trade-off between them. For example, virtual X-ray techniques (see Section 3.4) typically support the highest object interaction (containment), yet are not as scalable as other techniques with more modest strengths.

Domain: *proximity, intersection, enclosement, containment* (ordinal)

Characteristic Techniques:

- *proximity:* none
- *intersection:* view projection morphing [50], worlds-in-miniature [53], bird's eye views [58]
- *enclosement:* worldlets [53], BalloonProbe [44], 3D explosion probe [130]
- *containment:* image-space dynamic transparency [6], importance-driven volume rendering [140], view-dependent transparency [42]

3.3.3 Depth Cues

As we hinted at earlier in this thesis, actually relaxing some of the visual cues humans rely on for spatial perception will most certainly have a negative impact on the user's understanding of his or her surroundings, regardless of any advantages gained from doing this. The perception of depth, i.e. the actual 3D component of our vision system, is most vulnerable to this effect, and thus we define a dimension that captures the degree of depth cues that a technique provides.

Depth cues is an ordinal dimension with a five-value scale ranging from low to high, signifying the amount of depth cues retained by the technique; high would mean that in principle all depth cues are preserved, whereas low means that practically none are.

There are additional visual cues that help humans perceive their environment and that play a role in the classification of occlusion management techniques, some of which we capture in the "target invariances" dimension below.

Domain: *low, somewhat low, medium, somewhat high, high* (ordinal)

Characteristic Techniques:

- *low:* artistic multiprojection [1]
- *somewhat low:* 2D dynamic transparency [70], free-space transparency [79]
- *medium:* BalloonProbe [44], blueprints [106]
- *somewhat high:* image-based exploded view diagrams [92]
- *high:* tumbler [116], worldlets [53]

3.3.4 View Paradigm

Different occlusion management techniques utilize the view and the view space in different ways; this dimension captures the paradigm employed for managing the visual substrate. Typically, interaction techniques are either based on a single view, twin views, or a large number of views (multiple); similarly, for the case when there are additional views beyond the main one, they may either be separate windows in an overview+detail approach, or integrated in the same image in a focus+context [59] way. The view paradigm dimension is used to classify techniques according to a combination of these two metrics.

The degree of integration can sometimes be tricky to assess—for example, in the case of the worlds-in-miniature (WIM) [132] technique, there is a very obvious second view, i.e. a miniature version of the world, yet since it is a first-class object in the environment, we classify it as being integrated. For bird's eye views [58], on the other hand, the secondary view is in a separate window, and is thus classified as having twin separate views. This factor is also the reason why separating the number of views from their integration is difficult; in the case of Singh's multiprojection techniques [124], the single view actually consists of multiple different cameras, non-linearly combined into one.

Domain: *single view, twin separate views, twin integrated, multiple separate, multiple integrated* (nominal)

Characteristic Techniques:

- *single view:* 3D explosion probe [130]
- *twin separate views:* bird's eye views [58]
- *twin integrated views:* worlds-in-miniature [132], view projection animation [50]
- *multiple separate views:* worldlets [53]
- *multiple integrated views:* virtual multiprojection cameras [124], looking glass (multi-user) [94]

3.3.5 Interaction Model

We are also interested in capturing the specific interaction model employed by each technique; some rely on active user intervention, exposing occluded content according to direct (free-space transparency [79]) or indirect (temporally controlled non-linear projections [125]) input from the user, whereas others employ a passive interaction approach to present the hidden objects directly to the user with no input necessary (dynamic transparency [6]). A third possible option is a hybrid model, where the technique has two or more distinct modes during which the interaction model changes between active and passive. An example of the latter is the way-finder [4] system, which first calculates a path through the world in an off-line phase, then allows the user to interactively explore the path.

One interesting effect of a passive mode is that a technique employing such an interaction model typically must have prior semantic knowledge about the targets the user considers important (and sometimes even an interest value for each target); an example is the importance-driven volume rendering technique by Viola et al. [140]. Active mode, on the other hand, puts these decisions in the hands of the user, providing for more flexible interaction.

Domain: *passive, hybrid, active* (nominal)

Characteristic Techniques:

- *passive:* image-space dynamic transparency [6], multiblending [8]
- *hybrid:* way-finder [4], path drawing for 3D walkthrough [77]
- *active:* perspective cutouts [33], SDM [32]

3.3.6 Target Invariances

The sixth and final primary dimension of our taxonomy describes the number of invariances preserved by the technique. A complement to the depth cues parameter above, target invariances describes how many of the following properties of the **targets** (not necessarily distractors) in the environment are retained:

- **Location.** Position and orientation of the target.
- **Geometry.** Shape and size of the target.
- **Appearance.** Color, texture and material of the target.

All of the above properties are all more or less important for visualization applications in 3D environments; for instance, for a simple 3D scatterplot, the location of each data point is vital for the data to be interpreted correctly, so an occlusion management technique designed for use with such data should definitely preserve this property. For color-coded tree hierarchies, such as for a 3D representation of a file system, it might make sense to displace location (as long as connectivity information is retained) but the appearance should not be altered.

The higher number of invariances a technique retains, the better it is, and so this is an ordinal dimension. However, as discussed in the introduction, our normal visual cues are often at an odds with understanding various properties of an environment (e.g. seeing all the targets despite occlusion), and thus this is an example of a classical trade-off decision specific to each technique. Often, designers can gain certain attractive properties by relaxing others, all depending on the particular application area of the technique.

Domain: *0-3 properties: location, geometry, appearance* (ordinal)

Characteristic Techniques:

- *location:* preserve: interactive cut-away and break-away views [43]; discard: 3D explosion probe [130]
- *geometry:* preserve: worlds-in-miniature [132]; discard: non-linear projection [1, 125]
- *appearance:* preserve: BalloonProbe [44]; discard: view-dependent transparency [42]

3.4 Design Patterns

We have classified the 25 techniques involved in our survey using our taxonomy; the result can be summarized in the parallel coordinate plot in Figure 3.1. We then study this body of classifications to see patterns and trends, using for instance hierarchical clustering mechanisms. This analysis yields five distinct and orthogonal archetypes of design, or *design patterns* [3], i.e. a generic and reusable solution to a commonly occurring problem within a specific context. The five patterns we have identified we call Multiple Viewports, Virtual X-Ray, Tour Planner, Interactive Exploder, and Projection Distorter. We will describe these in the following sections.

According to pattern lore, a design pattern has four essential elements: a name, a problem (already given), a solution, and the consequences of using the pattern. We use these elements in our discussion of each pattern. We also show the distribution of techniques implementing the pattern on the design space.

3.4.1 Multiple Viewports

The Multiple Viewports pattern (red in Figure 3.1) is characterized by a view paradigm based on two or more separate views, resulting in an overview+detail kind of layout. Instances of this pattern also tend to preserve most, if not all, invariances—the trick lies in the placement of the additional cameras, not manipulating the image seen from them. It is most effective for 3D environments that lend themselves to overviews, such as landscapes and structured buildings.

Furthermore, the interaction model tends to be active; no existing technique performs the automatic placement of cameras that would be necessary for passive interaction.

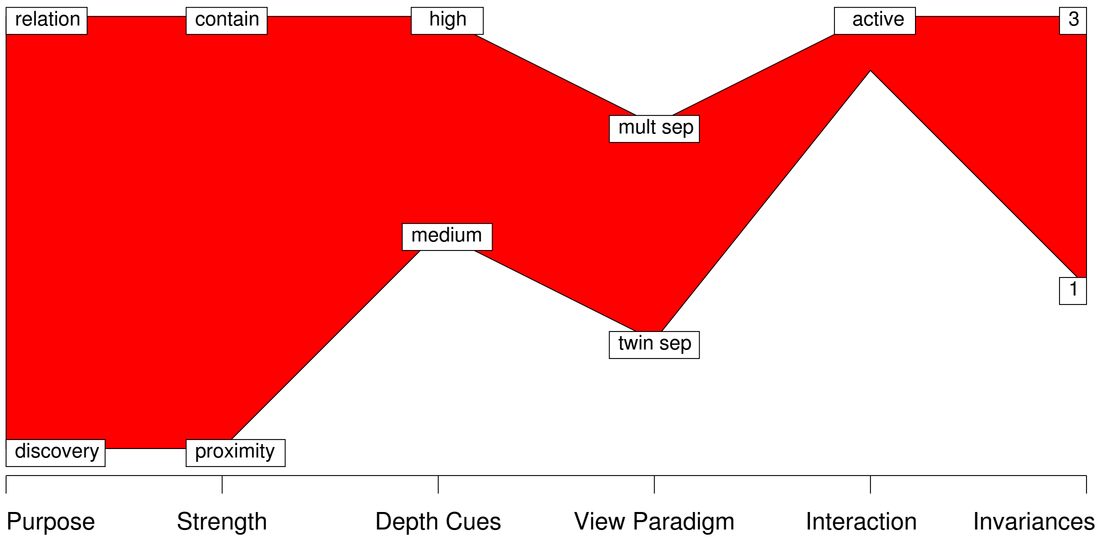


Figure 3.2: Classification distribution of multiple viewports techniques.

Solution: Manage discovery and access of targets by providing several alternate (often separate) viewports of the 3D environment. Typically, one viewport is designated as the main viewport, with the other viewports as secondary and generally smaller. Accordingly, the main viewport is often used for detail or first-person views, whereas the alternate views give either static or dynamic overviews of the environment (such as an overhead map).

Consequences: The use of the Multiple Viewports pattern trades screen estate and user attention for increased discovery and access; the user will have a smaller main visualization window than otherwise, and may have to split his or her attention across all of the viewports. Furthermore, in some situations, it is not clear what constitutes an overview, and thus introducing additional viewports may have diminishing returns. However, this is a very powerful approach for suitable environments.

Examples: Tumbler [116], worlds-in-miniature [132], worldlets [53], bird's eye views [58].

3.4.2 Virtual X-Ray

The Virtual X-Ray pattern (green in Figure 3.1) is based on an image-space approach where occlusion can be easily detected and sometimes even delegated to programmable fragment shaders. The pattern is not limited to 3D—the same idea permeates dynamic transparency techniques for 2D windowing systems, such as the free-space transparency [79] and multiblending [8] techniques. Typically, example techniques have very high disambiguation strength. Furthermore, there is clear division between two types of Virtual X-Ray techniques: active ones, where the user controls a “searchlight” on the 2D view, and passive ones, where semantic information allows the system to automatically uncover targets.

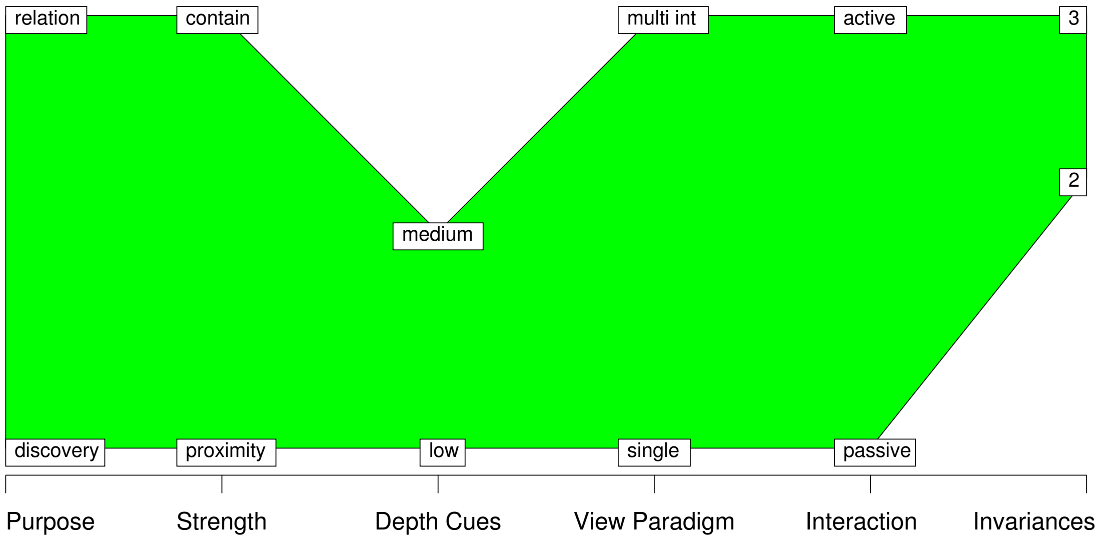


Figure 3.3: Classification distribution of virtual X-Ray techniques.

Solution: Make targets visible through intervening distractors by turning occluding surfaces invisible or semi-transparent. The method for distractor removal is characteristic: some techniques are view-dependent (breakaway) whereas others are static (cutaway); some eliminate distractors (or parts of distractors), others merely make distractors semi-transparent. Active interaction facilitates exploration whereas passive interaction requires target information but yields a potentially higher correctness.

Consequences: The Virtual X-Ray pattern makes discovery trivial and facilitates access by selectively removing distractors occluding the targets. However, this is a direct weakening of occlusion depth cues, causing a decrease in depth perception and making spatial relation more difficult. The use of semi-transparency also results in high visual complexity and imposes a high cognitive load on the user. Finally, Virtual X-Ray can make visibility computations for rendering optimization useless.

Examples: Perspective cutouts (active) [33], image-space dynamic transparency (passive) [6].

3.4.3 Tour Planner

The family of Tour Planner techniques (blue in Figure 3.1) is characterized by a hybrid interaction model consisting of an offline and an online phase where first the path is defined or computed and then interactively shown in the environment itself. Typically no distortion is imposed on the view (a temporal canvas is used), so all invariances are usually retained.

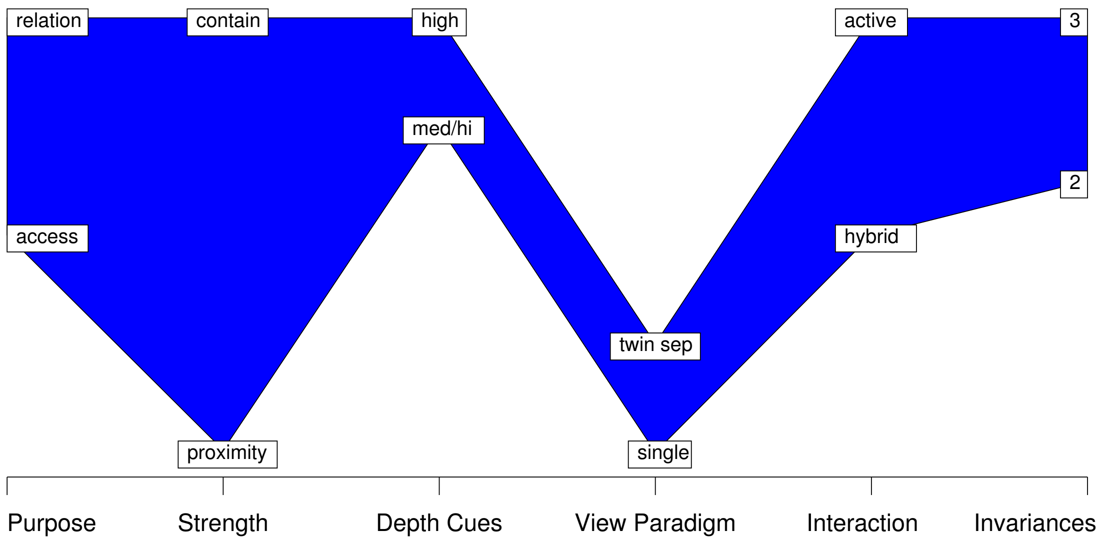


Figure 3.4: Classification distribution of tour planner techniques.

Solution: Present all targets in an environment by constructing a complete (i.e. all targets are visible in at least one point) path through it. It should also conform to a number of additional constraints (such as short or optimal length, closed, uniform visual complexity, etc). Often realized in an offline precomputation or specification step followed by an interactive exploration phase where the user is guided by the computed path.

Consequences: The Tour Planner pattern is non-invasive and thus will not modify the environment itself and will typically retain all invariances. This however means that the pattern's disambiguation strength is generally low. The path computation step can sometimes be costly in terms of computation time, and intractable to dynamically changing situations.

Examples: Way-finder [4], 3D navigation guidance [49].

3.4.4 Interactive Exploder

Interactive Exploders (purple in Figure 3.1) manage occlusion in the object space through active user interaction in a direct manipulation approach. The exploding metaphor means that target location is rarely retained, although most other invariances typically are.

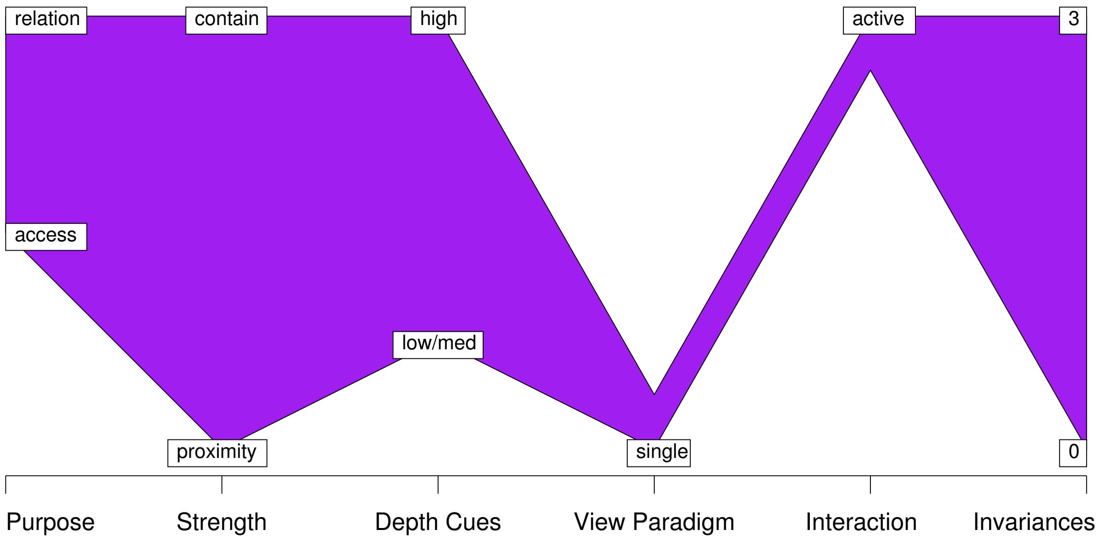


Figure 3.5: Classification distribution of exploder techniques.

Solution: Provide a user-controlled distortion probe that locally displaces objects to manage occlusion. The approach is based either on (i) removing distractors or (ii) separating targets; in the former case, we want to eliminate objects that get in the way, whereas in the latter, we instead want to disambiguate between several targets who share the same space. The effect is similar to that of exploding diagrams used for technical illustration, but here the interaction is active and under direct user control in the object space. Implementations often provide some kind of visual cue of the original positions of displaced objects, typically using wireframe or ghosting (transparency).

Consequences: Using an Interactive Exploder can help disambiguate even very difficult situations, but the very nature of the pattern means that at the very least location is not preserved. The pattern is best suited for discovery. The local influence model means that there may be a problem of reach in a virtual environment.

Examples: 3D explosion probe [130], deformation-based volume explosion [102].

3.4.5 Projection Distorter

This pattern (black in Figure 3.1) is signified by a view-space approach presented using two or more integrated views. Since non-linear projections are typically employed to pack as many of the targets as possible into a single view, few invariances are retained. Thus, this pattern is often best used for discovery, rarely for access, and almost never for relation.

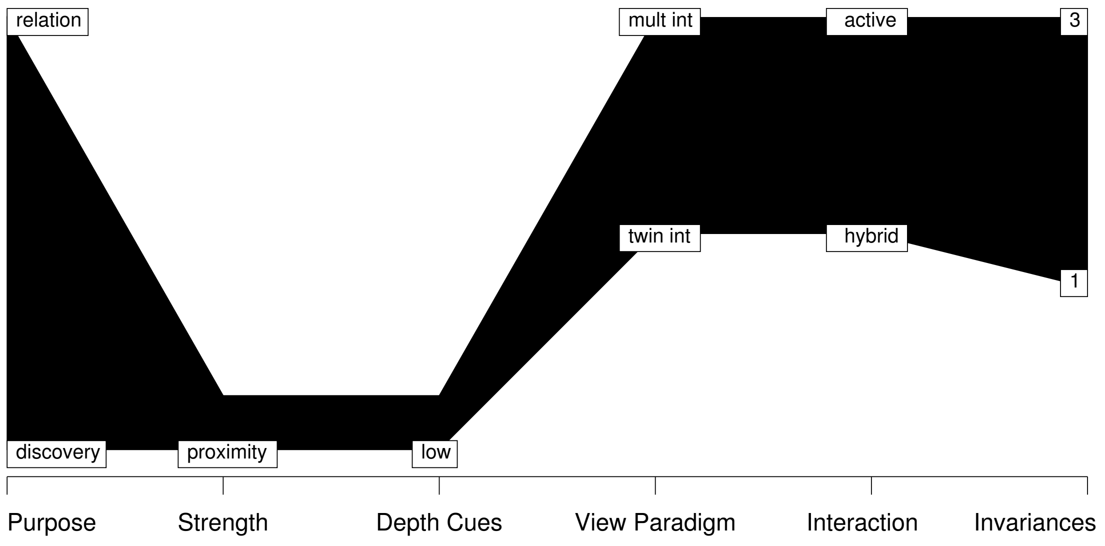


Figure 3.6: Classification distribution of projection distorter techniques.

Solution: Integrate several different views of targets into a single view in order to maximize discovery. The solution is then often reminiscent of a focus+context technique with one focus per view. Individual view selection is often actively controlled by the user in an online or offline manner. In one case, a hybrid approach is employed where target semantic information is extracted from previous user explorations using data mining techniques and then used to inform the technique [125].

Consequences: The use of the Projection Distorter pattern affects only the view projection code of an application and is thus relatively easy to integrate into existing code. On the other hand, the resulting visual displays can often become disconcerting and disorienting to the user. Few object properties are retained.

Examples: Artistic multiprojection [1], view projection animation [50].

3.5 Future Research Directions

Besides identifying existing design archetypes in the literature, we can also extract possible future research opportunities from our taxonomy by studying the as-of-yet unexplored parts of the design space. One such observation is the need for techniques that make users aware of occluded content without compromising visual quality and imposing a high cognitive load on the user. Retaining a high degree of depth cues is important for complex visual tasks such as spatial relation. Another interesting area to explore is hybrid-interaction methods where the user's own actions are used to inform the target selection. This approach may help solve the trade-off between the precision that a passive interaction model provides as opposed to the more general nature of active user interaction.

Combinations of patterns could be profitable ways of utilizing the strong points of two different methods while at the same time making up for the weak ones. For example, a multiple-viewport technique could be augmented with virtual X-Ray support in one or several of the views. A tour planner could be paired with an interactive exploder to help disambiguate in difficult situations of locally high target congestion.

General 3D navigation has been shown to be a task with a very high cognitive load; for every traveled world unit, the user runs the risk of becoming disoriented, totally lost, or even nauseous. For the longer term, it can be noted that the ultimate goal of occlusion management techniques should be to help minimize the need for 3D navigation in general. Perhaps the class of interaction techniques described in this paper can help short-circuit excessive navigation in the first place.

3.6 Discussion

The taxonomy presented in this paper has been designed to be orthogonal and objective, with no dimension being reducible to another and having a minimum of coupling to the other dimensions. Regardless, it is always possible to debate the inclusion or exclusion of specific property dimensions to a taxonomy. We believe this to be a valid one, and the successful classification of 25 different techniques using it confirms this claim.

Nevertheless, property dimensions that were excluded for various reasons include scalability (the amount of object density the technique can handle), influence level (i.e. whether the interaction technique operates on a local, regional, or global level), and dimensionality (2D, 2.5D, 3D, etc). Classification using these and other dimensions is left as an exercise to the reader.

Despite the lofty goal mentioned in the previous section of bypassing the need for navigation, a number of 3D navigation techniques were indeed included in the classification in this paper. These were selected due to them being on the borderline of what constitutes an occlusion management technique, or representative for a specific class of techniques. Many other 3D navigation techniques in the literature were excluded from this classification; the line had to be drawn somewhere.

An interesting observation on the design patterns identified in this paper is that the separating feature between most patterns is the approach taken to visualize occlusion (the canvas used, if you will). For instance, virtual X-Ray techniques use image space, projection distorters use view space, interactive exploders use object space, and tour planners utilize temporal space.

3.7 Conclusions

Occlusion management is a subset of 3D interaction techniques concerned with improving human perception for specialized visual tasks through manipulation of visual cues such as occlusion, size, and shape. In this paper, we have presented five archetypical design patterns for occlusion management based on a classification of existing interaction techniques. The patterns include multiple viewports, virtual X-ray, tour planners, interactive exploders, and projection distorters. The underlying taxonomy used for this classification is based on six characteristic properties of occlusion management techniques. Analysis of this taxonomy also yields additional missing patterns, such as primarily techniques for target awareness and hybrid-interaction approaches with an emphasis on retaining a high degree of depth cues and supporting spatial relation.

Chapter 4

View Projection Animation[1]

Niklas Elmqvist[2], Philippas Tsigas[2]

Abstract

Inter-object occlusion is inherent to 3D environments and is one of the challenges of using 3D instead of 2D computer graphics for information visualization. Based on our general theoretical framework of 3D occlusion, we present an interaction technique for view projection animation that reduces inter-object occlusion in 3D environments without modifying the geometrical properties of the objects themselves. The technique provides smooth on-demand animation between parallel and perspective projection modes as well as online manipulation of view parameters, allowing the user to quickly and easily adapt the view to avoid occlusion. A user study indicates that the technique significantly improves object discovery over normal perspective views. We have also implemented a prototype of the technique in the Blender 3D modeler.

Keywords: occlusion management, 3D visualization, view projection

4.1 Introduction

Three-dimensional computer graphics provides considerable potential for information visualization. However, there is an increased overhead associated with using 3D over conventional 2D graphics for the purposes of orientation and navigation within an environment. In general, 3D graphics imposes a high cognitive

[1]This is an extended version of a paper that appeared in *Proceedings of the ACM Conference on Advanced Visual Interfaces 2006*, Venice, Italy.

[2]Department of Computer Science, Chalmers University of Technology, 412 96 Göteborg, Sweden. E-mail: {*elm,tsigas*}*@cs.chalmers.se*

load on users trying to gain and maintain an overview of the environment, and often cause disorientation, confusion, and sometimes even nausea. One of the central issues behind this effect is *occlusion*, the phenomenon that nearby objects occlude more distant objects in 3D even if the objects are not overlapping in space.

Why is occlusion a problem in 3D visualization environments? There are three basic issues. First, and perhaps most importantly, there is a *discovery* problem if an object is occluded since then the user may never know that it exists. Secondly, even if the user is aware of the existence of an occluded object, there is an *accessibility* problem, since the user will have to move the viewpoint in some nontrivial way in order to retrieve the information encoded in the object. Finally, even if the user is able to discover and access individual occluded objects in the world, the high-level task of spatially *relating* the objects to each other can be difficult.

In this paper, we develop an interaction technique for view projection animation that aims to reduce inter-object occlusion in 3D environments without modifying the geometrical properties of the environment itself, nor the objects in it. The technique allows for smooth animation between the conventional perspective projection mode, which mimics human vision in the real world and is commonly used for 3D visualizations, and parallel projection mode, where the depth coordinate is ignored and objects are assigned screen space according to their actual geometrical size, regardless of their distance to the viewpoint.

A formal user study conducted on our technique in relation to traditional perspective projection shows a significant improvement of object discovery in 3D environments in our favor. The cost for this increased efficiency is instead significantly longer task completion times; users essentially trade speed for accuracy when using our technique. On the other hand, the results also show that there is no statistically significant difference in completion times between using our projection animation technique and a user-controlled camera. In fact, we believe that our projection animation technique requires fewer viewpoint position manipulations than normal perspective views, and thus users run a lower risk of becoming disoriented when navigating 3D space.

This paper begins with a review of existing work in the area. We then launch ourselves into a theoretical treatment of the occlusion problem and its human-computer interaction aspects, mapping out the problem space and its components. After that, we describe the projection animation technique itself. This is followed by an overview of the formal user study we conducted, for both exploring the occlusion problem as well as comparing our new technique to normal perspective views, and the results we collected from it. We close the paper with a discussion and some conclusions.

4.2 Related Work

While most work on improving the usability of 3D visualizations attacks the higher-level problem of navigation, there also exists a number of papers dealing more directly with object discovery and access in complex 3D environments. The Worlds-in-Miniature technique [132] uses a miniature 3D map of the environment to support both discovery and access, worldlets [53] provide both global overview maps as well as local views optimized for detail, bird's eye views [58] combine overview and detail views of the world, and balloon force fields [44] inflate to separate occluding objects. None of these makes direct use of the view projection to improve perception; however, temporally controlled non-linear projections [125] have been used to great effect in improving navigation and perception of 3D scenes.

Projections are intrinsic to computer graphics, but are mostly limited to linear perspective projections. CAD programs have traditionally made use of parallel projection, often through multiview orthographic projections where two or more views of the same object are shown on planes parallel to the principal axes. Carlbom and Paciorek [23] (see also [103]) give a complete overview of planar geometric projections and their various advantages and disadvantages in different situations.

Recent developments in the area also include multiprojection rendering, where several perspectives are combined into one, mainly for artistic purposes. Agrawala et al. [1] compose views of multiple cameras, where each object is assigned to a specific camera perspective, allowing for creative scene composition akin to the work of famous painters. Singh [124] uses a similar approach, but smoothly combines the multiple viewpoints into an interpolated virtual camera in view space instead of composing the images of disjoint cameras on an image-level. While only slightly related to our technique, these works give valuable insight into the manipulation of projection transforms. Our technique can also be compared to glances [111], with the perspective view as the base view and the parallel view as the glance.

Our projection animation technique allows for interactive manipulation of camera properties beyond the traditional position and orientation parameters. Prior work in this area includes the IBar [126] camera widget, which is a comprehensive camera control that provides intuitive ways of manipulating these parameters for the purposes of 3D scene composition. Another approach uses image-space constraints to solve for the camera parameters given a number of control points [64]. Our work is again focused on reducing the impact of objects obscuring other objects rather than direct camera control, and the camera manipulations provided by our technique only give additional means to achieve this.

Finally, the view projection animation technique described in this paper bears close resemblance to the *orthotumble* technique presented by Grossman et al. [69]

and its predecessor, the animated view transitions described in [68], but the purpose of these are primarily for maintaining and understanding 3D models rather than reducing occlusion. In addition, our approach uses a more correct algorithm for the "dolly-and-zoom" effect, whereas the orthotumble algorithm is based on linear matrix interpolation.

4.3 Analysis

In this paper, we present a specific occlusion management technique called view projection animation where the camera projection matrix is smoothly animated from perspective to parallel projection and back. We can categorize this technique as part of a more general solution strategy based on dynamic manipulation of the view projection to favorably present objects and minimize occlusion in the environment.

View projection animation clearly improves object discovery by providing the user with means to avoid nearby objects hiding more distant ones. Toggling between the projection modes yields two different perspectives on the environment as well as intervening views during the smooth animation between them. This strongly facilitates visual search and, to some extent, target identification, by disambiguating between occluding objects. Object access benefits much less from the technique; previous knowledge of the target's location is of little use when the view space is non-linearly transformed by the technique.

The applicability of the technique is limited to intersecting objects: since we do not transform the space itself, enclosed and contained objects will remain occluded even after the projection transformation. As will be seen in the user study, the technique performs well at low to medium-sized object density.

4.4 View Projection Animation

The idea behind our technique for view projection animation is to combine the major virtues of parallel and perspective projections: that (i) perspective projections offer a realistic view of a 3D environment akin to our perception of the real world, and that (ii) parallel projections offer a more accurate and exact view of the environment. Furthermore, the nature of parallel projection means that inter-object occlusion is reduced in comparison to perspective projection since objects are assigned screen space according to their geometrical size only, regardless of their distance to the camera. Using perspective projection, a tiny object can fill the whole viewport if located sufficiently close to the viewpoint.

By combining these two projection modes into the same interaction technique, we are potentially able to enjoy the best of both worlds: the view defaults to perspective projection when the user is navigating the space normally, but allows for easy switching (glancing) to parallel projection when the user needs to perform object discovery or access. Furthermore, the transition between perspective and parallel projections, and vice versa, is smoothly animated to allow the user to maintain context of the environment and the projection at all times with a minimum of cognitive overhead. The transition also provides additional information on the structure of the 3D scene.

In addition to transitions back and forth between perspective and parallel projections, we augment our technique with functionality to change the center of projection as well as to modify the field-of-view angle in the perspective projection mode. Changing the center of projection gives an additional means for the user to arbitrate between occluding objects, and by gaining control of the field of view, the user can smoothly zoom in and out of the 3D environment at will. See Figure 4.1 for more details.

For our interaction technique, we define three input buttons, labeled PROJ, UTIL, and SHEAR, respectively; these can be mouse or keyboard buttons. In addition, the technique also captures mouse motion for some parameter changes, notably the field-of-view and center-of-projection (direction-of-projection for parallel mode) modification states. The parallel projection mode has a number of pre-defined oblique projection modes that the user can cycle between: orthographic (head-on parallel projection) versus cavalier and cabinet projections, where the direction of projection is set at fixed values. Note that for all parallel projection modes, the release of the PROJ input button will smoothly revert the projection back to the default perspective mode. Reverting to the default state will also reset all view parameters, such as centering the center (or direction) of projection and setting the focal length to the default value.

4.4.1 Projection Transitions

Transitions between various projection states are performed through simple linear interpolation between the source and destination projection transforms. In the case of the parallel-to-perspective transition (and its inverse), however, a linear interpolation will yield unsatisfactory results due to the non-linear relation between these two projections. For this case, we need to explore the matrix M that relates the two projection matrices P_{par} and P_{per}.

As discussed above, a parallel view transform represents the situation where the focal length of the camera is infinite. The transition from perspective to parallel view can be approximated in a real-life camera by a so-called “dolly and zoom” operation, where the camera is moved backwards at the same time as the focal length is increased (i.e. zoomed in). By keeping these parameters balanced, the focused object in the camera view will maintain the same size and shape, but the rest of the scene will appear to be “flattened”. We simulate this effect in our transition between perspective and parallel projection.

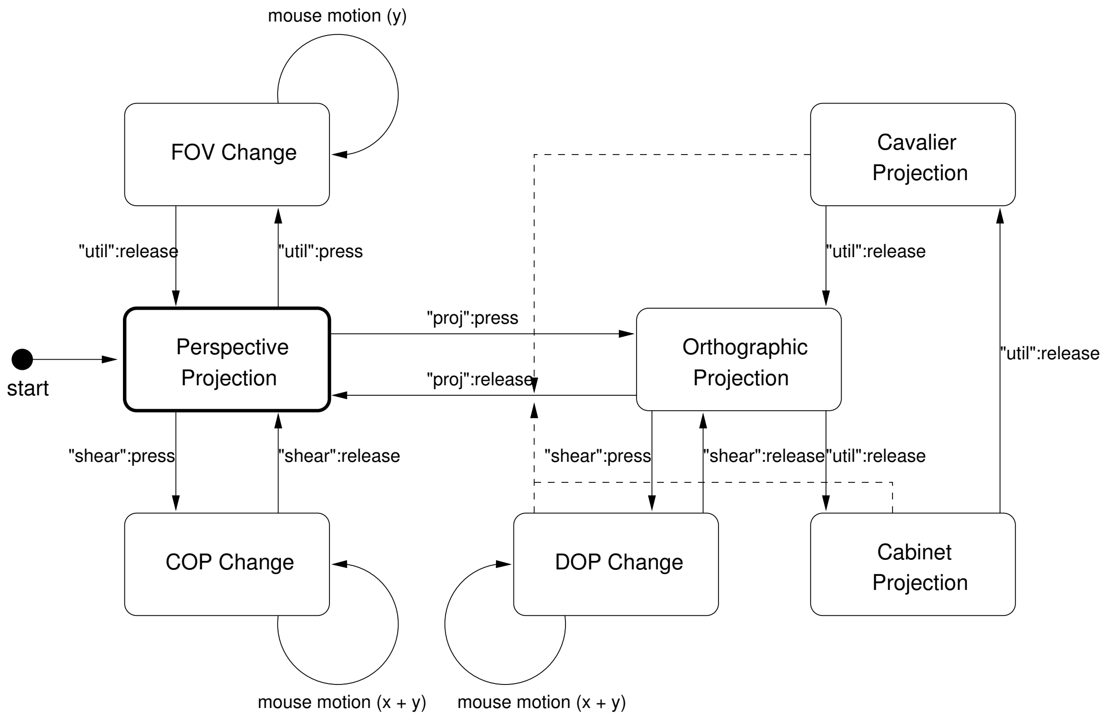


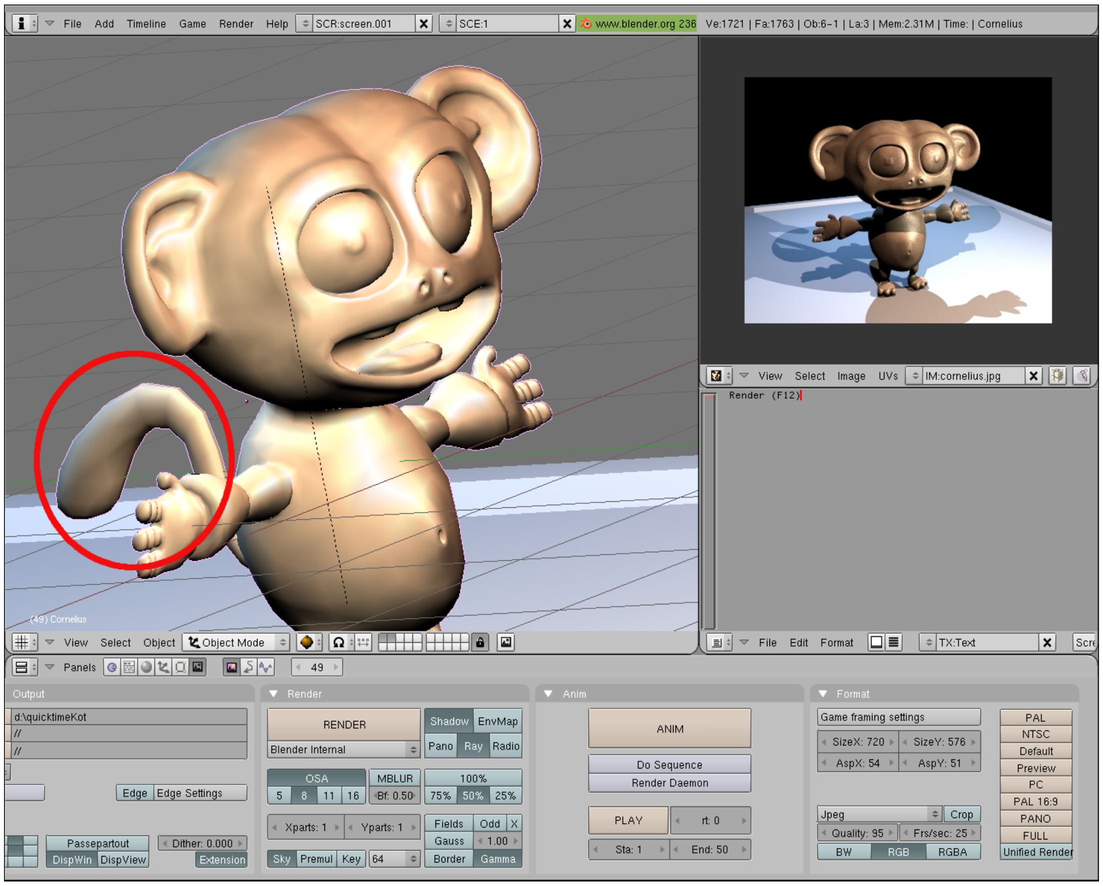
Figure 4.1: State diagram for the projection animation interaction technique.

4.4.2 Implementation

We have implemented our interaction technique in a sample C++ application called PMorph. This application consists of a $100 \times 100 \times 100$ unit-sized cube populated with n boxes with randomized geometrical and graphical properties. The 3D rendering is performed using OpenGL. The application provides mouse-driven view controls with a camera that can be orbited and zoomed in and out around a focus point in the center of the environment. The implementation of the interaction technique itself hooks seamlessly into the input handlers of the windowing system and requires no additional modification to the implementation of the 3D environment or the 3D objects.

4.4.3 Case Study: Blender Implementation

In order to study the feasibility and flexibility of our projection animation technique, we also implemented it inside the Blender [15] 3D modeling package. Blender is a very powerful and widely used 3D software suite that is freely available as Open Source under the GPL license. Our implementation integrates seamlessly into Blender and allows modelers to animate between parallel and perspective projections in different 3D windows. The software naturally already



Ve:1721 | Fa:1763 | Ob:6-1 | La:3 | Mem:2.31M | Time: | Cornelius

Render (F12)

(49) Cornelius

49

d:\quicktimeKot

Blender Internal

5 8 11 16 Bf: 0.50

100% 75% 50% 25%

Xparts: 1 Yparts: 1 1.00 64

Dither: 0.000

rt: 0

Sta: 1 End: 50

SizeX: 720 SizeY: 576

AspX: 54 AspY: 51

Jpeg

Quality: 95 Frs/sec: 25

Figure 4.2: Blender implementation screenshot. Note the parallel view (left) showing geometric features not visible in the perspective view (right).

supported these projection modes prior to our modifications, so we changed the projection code to perform a smooth animation between the two matrices. In addition to this, we introduced the capability for users to change the center of projection while in orthographic mode, providing an additional way to reduce occlusion. Figure 4.2 shows a screenshot of the modified Blender software.

While a 3D modeler is not the primary target platform for our technique (even though Grossman et al. [68, 69] use the effect for this very purpose), this case study shows that the technique can indeed be implemented seamlessly inside existing 3D applications with only small modifications to the old code.

4.5 User Study

We have conducted a formal user study with two main motivations: (i) to empirically investigate the impact of occlusion on object discovery efficiency in 3D environments, and (ii) to study the performance of users given access to our view projection animation technique in comparison to users with a normal perspective view.

4.5.1 Subjects

We recruited 26 subjects, six of which were female, from the undergraduate engineering programs at our university. No previous experience of 3D applications was required. Ages ranged from 20 to 40 years of age, and all subjects had normal or corrected-to-normal vision.

4.5.2 Equipment

The experiment was conducted on a Pentium III 1 GHz desktop PC with 512 MB of memory and running the Linux operating system. All tasks were carried out using our prototype implementation. The display was a 19" monitor with the main visualization window fixed at 640×480 size.

4.5.3 Task

Subjects were asked to perform object discovery in a simple $100 \times 100 \times 100$ environment filled with 3D boxes by counting the number of boxes of a given color. Target colors were restricted to one of the primary RGB colors (i.e. red, green, or blue), and all distracting objects were configured to contain no elements of that color component. Each task instance was fully randomized, including the position, orientation, and size of the distractors. At least 1 and at most 10% of the total number of objects were targets. Box dimensions (both targets and distractors) ranged from 1% to 12.5% of the environment dimensions. Intersection but no enclosement or containment was allowed. A simple 20×20 line grid was rendered at the bottom of the environment to facilitate user orientation.

The camera focus point was fixed at the center of the environment and the orientation was randomized within 60° from the horizontal. In addition, the camera position was also randomized and offset sufficiently from the focus point so that all objects in the scene were visible. Field-of-view angle for the perspective view was fixed at 60°. For the dynamic camera, the users could freely orbit the camera around the focus point as well as change the focus distance.

4.5.4 Design

The experiment was designed as a repeated-measures factorial ANOVA, with the independent variables Density (two levels, "low" or "high"), Camera ("static" or "dynamic", i.e. a fixed or a user-controlled camera), and PMorph ("on" or "off", i.e. whether the projection animation technique was available or not), all of them within-subjects. The dependent variables were the number of found target objects and the completion time for each task. Subjects received the PMorph and Camera conditions in randomized order to avoid systematic effects of practice; for the Density condition, the ordering was low to high.

Users performed the test in sets of 10 tasks for each condition. Each task scenario was completely randomized, with either 50 or 200 total objects in the environment depending on the density, and up to 10% of them being targets.

For each specific condition, subjects were instructed in which features (dynamic or static camera, projection animation on or off) were available to them. Tasks were given automatically by a testing framework implemented in the software and answers were input by the user directly back into the framework, thus requiring no intervention by the test administrator. The software silently recorded the completion time, total target number, and found target number for each task. Trial timing started as soon as each new task was presented, and ended upon the subject giving an answer.

Each session lasted approximately thirty to forty minutes. Subjects were given a training phase of up to five minutes to familiarize themselves with the controls of the application.

With 26 participants and 10 search tasks for each of the 8 conditions, there were 2080 trials recorded in total. After having completed the full test, subjects were asked to respond to a post-test questionnaire.

4.6 Results

We divide the results from the user study into completion times, correctness, and subjective ranking categories. Note that for the correctness measure, we derive the cumulative errors for each task set from the sum of the differences between the total number of targets and the found targets for each task. The error ratio is defined as the cumulative error divided by the sum of the total number of targets for the task set, i.e. the number of errors per target.

4.6.1 Time

The mean completion time of solving a full object discovery task set (10 tasks) using normal perspective projection was 128.093 (s.d. 7.803) seconds, whereas the mean completion time for projection animation was 162.311 (s.d. 9.697) seconds. This is also a significant difference ($F(1, 25) = 38.752, p < 0.001$).

Figure 4.3 summarizes the results for the individual density conditions. It is worth mentioning that the difference in average completion times for normal perspective views versus projection animation using a dynamic camera in high density was not significant: 246.778 (s.d. 22.461) versus 273.745 (s.d. 20.998) seconds, respectively.

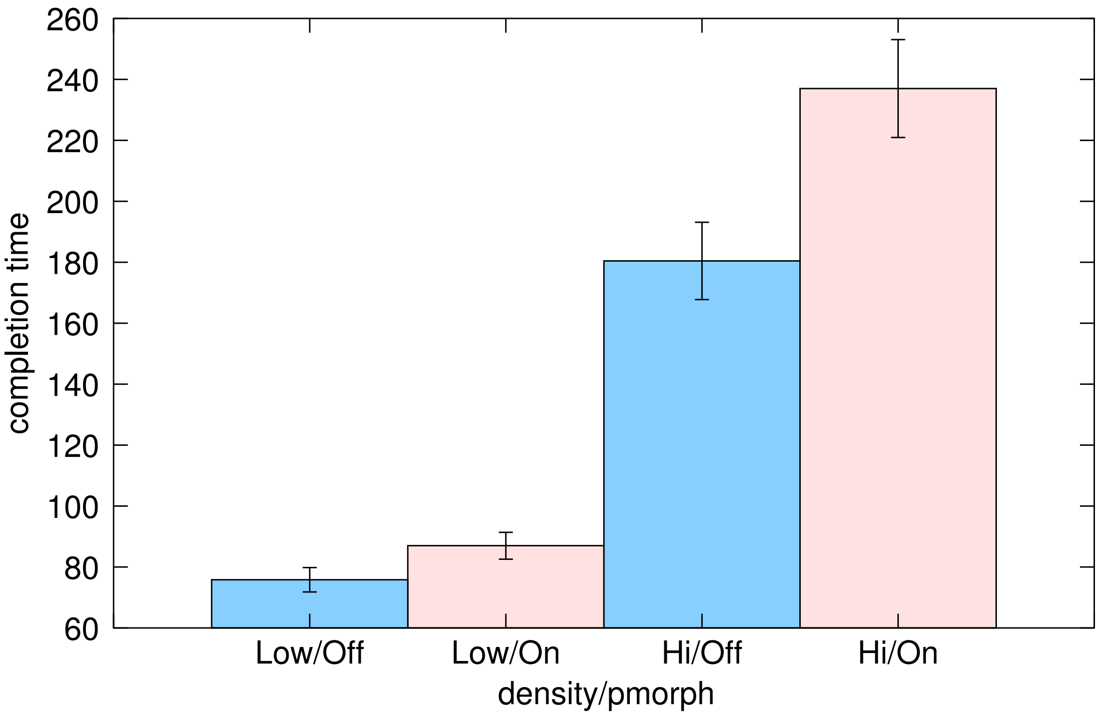


Figure 4.3: Mean completion times for solving a full task set for both camera types combined (standard deviations shown as error bars).

4.6.2 Correctness

The average error ratio for a full task set (10 tasks) using normal perspective projection compared to projection animation was 0.095 (s.d. 0.003) versus 0.055 (s.d. 0.003), respectively. This is a statistically significant difference ($F(1, 25) = 75.757$, $p < 0.001$). Not surprisingly, density had a significant impact on correctness ($F(1, 25) = 407.290, p < 0.001$); the average error ratio for the low density condition was 0.022 (s.d. 0.002), to contrast with 0.127 (s.d. 0.005) for the high density condition. This suggests that occlusion does negatively affect object discovery efficiency.

See Figure 4.4 for an overview of additional correctness results. One notable fact is that the difference in average error ratio for projection animation versus normal perspective in the dynamic camera condition was not significant: 0.081 (s.d. 0.008) versus 0.080 (s.d. 0.008) ($F(1, 25) = 0.006, p = 0.941$).

4.6.3 Subjective Rankings

The rankings given by the participants in the post-test questionnaire are overall positive and in favor of our projection animation technique; see Table 4.1 for

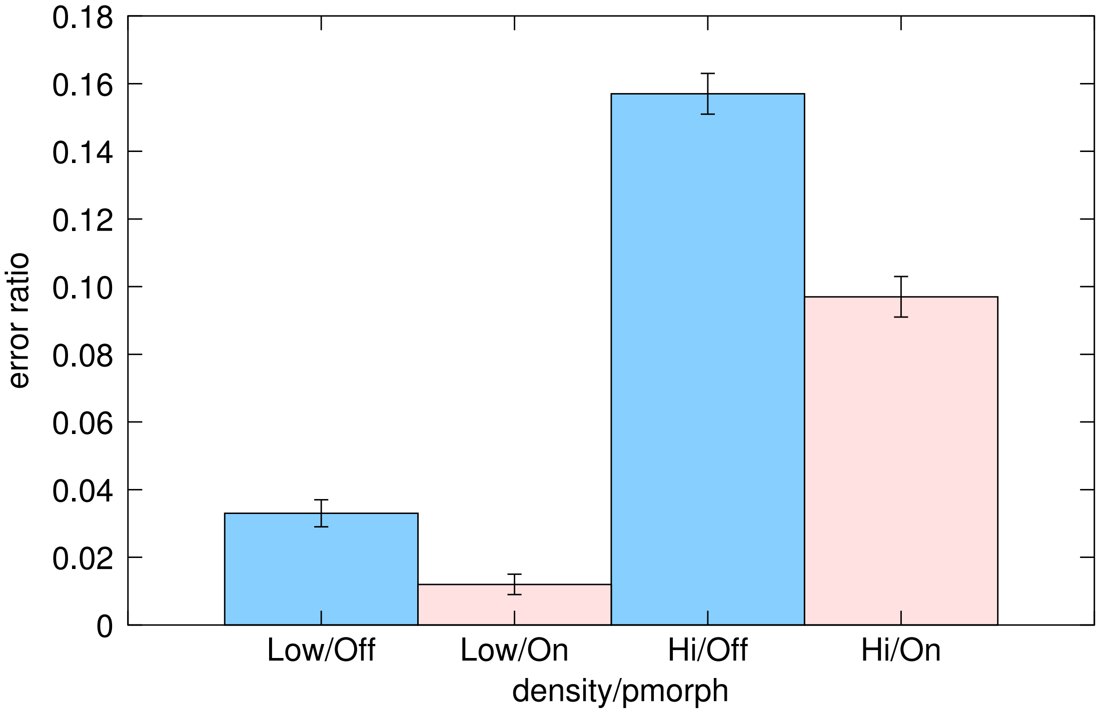


Figure 4.4: Mean error ratios for solving a full task set for both camera types combined (standard deviations shown as error bars).

an overview. The Q2 and Q6 questions, relating to perceived efficiency and preference, are of special interest, and are significantly in favor of our technique (65% and 73%, respectively).

4.7 Discussion

This paper presents two main contributions: (i) the analysis of the space of the occlusion problem in 3D environments, and (ii) the view projection animation technique used to reduce inter-object occlusion for any 3D visualization. Both of these contributions are validated by the results of the formal user study; we see that increasing object occlusion leads to significantly reduced discovery efficiency, and that the availability of projection animation significantly boosts efficiency in all object density conditions, respectively. In addition, by giving users control over the viewpoint, the impact of the occlusion problem is significantly diminished. On the other hand, this comes at the cost of longer completion times; the participants spent much more time solving tasks when having access to projection animation or a controllable camera, essentially trading speed for accuracy.

It is particularly interesting to study whether a user-controlled camera is suf-

Task	**Description**	**Normal**	**PMorph**	**Undecided**
Q1	Ease-of-use	58%	35%	7%
Q2	Efficiency	23%	65%	12%
Q3	Enjoyment	19%	69%	12%
Q4	Confidence	23%	69%	8%
Q5	Speed	50%	46%	4%
Q6	Overall	15%	73%	12%

Table 4.1: Post-test ranking results.

ficient to negate the occlusion problem, and whether the projection animation technique presented here is necessary. There is no clear benefit of projection animation over a traditional dynamic camera. However, we claim that projection animation is orthogonal to controllable cameras, and that they complement each other. Furthermore, our informal observations during the user study indicated that users with access only to a controllable camera performed significantly more view changes than when having access to both a controllable camera and projection animation. All 3D view changes incur a risk of loss of context and orientation, especially for high object densities, and so it is in our best interest to keep the amount of such changes low. We suggest that a combination of the two conditions will work best for practical applications.

Parallel projection assigns screen space to objects proportional to their geometrical size regardless of the distance to the camera, but the drawback is that the viewing volume is a box instead of a pyramidal frustum as for perspective projection. This means that peripheral objects will be lost in parallel mode, essentially rendering these objects impossible to discover. By smoothly combining *both* parallel and perspective projection into a single interaction technique, we are able to sidestep this problem and get the best of both worlds from the two projections.

A potential drawback of the technique is that the use of parallel projection leads to a loss of some depth cues in a 2D image of the environment (more specifically relative and familiar size as well as relative height). However, the spring-loaded nature of the interaction allows users to switch easily back and forth between projection modes to disambiguate potential depth conflicts between objects.

Finally, it might be argued that the other features of the interaction technique that do not relate to the projection animation are superfluous. However, while we did not include this functionality in the user study, we claim that these features are useful for occlusion reduction, and represent parameters that are intrinsic to the projection matrix of a camera, and thus in spirit with the projection animation interaction technique as a whole. Nevertheless, it is certainly possible to isolate merely the projection animation mechanism and implement only this in an external application.

4.8 Conclusions

We have presented an interaction technique for the seamless integration of perspective and parallel projection modes, allowing users to combine realism with accuracy, as well as reducing inter-object occlusion in 3D environment views. Results from a user study conducted on a prototype version of the technique show that occlusion in 3D environments has a major impact on efficiency, but that our technique allows for significant improvements in both object discovery and object access. Our technique treats 3D objects as immutable entities and requires no changes to the implementation or representation of the 3D environment, and should thus be possible to integrate with almost any 3D visualization.

Chapter 5

Interactive 3D Space Distortion[1]

Niklas Elmqvist[2]

Abstract

Using a 3D virtual environment for information visualization is a promising approach, but can in many cases be plagued by a phenomenon of literally not being able to see the forest for the trees. Some parts of the 3D visualization will inevitably occlude other parts, leading both to loss of efficiency and, more seriously, correctness; users may have to change their viewpoint in a non-trivial way to be able to access hidden objects, and, worse, they may not even discover some of the objects in the visualization due to this inter-object occlusion. In this paper, we present a space distortion interaction technique called the BalloonProbe, which on the user's command inflates a spherical force field that repels objects around the 3D cursor to the surface of the sphere, separating occluding objects from each other. Inflating and deflating the sphere is performed through smooth animation, ghosted traces showing the displacement of each repelled object. Our prototype implementation uses a 3D cursor for positioning as well as for inflating and deflating the force field "balloon". Informal testing suggests that the BalloonProbe is a powerful way of giving users interactive control over occlusion in 3D visualizations.

Keywords: occlusion reduction, 3D space distortion, interaction technique

[1]This is an extended version of a paper that appeared in *Proceedings of the ACM Symposium on Virtual Reality Software & Technology 2005*, Monterey, USA.

[2]Department of Computer Science, Chalmers University of Technology, 412 96 Göteborg, Sweden. E-mail: *elm@cs.chalmers.se*

5.1 Introduction

There is strong potential for making use of advances in the field of information visualization inside 3D virtual environments, where users can effortlessly overview and interact with the visualized data using natural interaction methods and multimodal techniques. However, the immersiveness of a virtual environment may also be a hindrance to the effectiveness of using the visualization; objects in a 3D environment will inevitably occlude other objects, partially or totally, leading to a phenomenon of literally not being able to see the forest for the trees. This inter-object occlusion problem applies to all 3D visualization, and can either result in reduced performance by forcing the user to change the viewpoint to access hidden objects, or even in reduced correctness due to some objects remaining undiscovered by the user.

The intuitive approach employed in this paper is to equip the user with a portable 3D force field that can be inflated and deflated to separate occluding objects, much akin to a balloon that pushes nearby objects away from each other as it is inflated. This interaction technique, accordingly called a BalloonProbe (see Figure 5.1 for an example screenshot), is freely controlled by the user in 3D space and is smoothly animated when it is inflated and deflated. Ghosted traces of the original position of the displaced objects are drawn to help the user maintain a cognitive overview of the original 3D scene. The displaced objects can optionally be subject to collision detection to avoid overlaps on the surface of the balloon, but this is only necessary for special cases when an object is fully contained by another.

We have implemented a prototype of the BalloonProbe technique on a standard PC using a mouse and keyboard for input, as well as a low-budget 6DOF motion tracker. Informal testing using a few subjects indicate that the BalloonProbe is a very useful tool for disambiguating complex 3D visualizations with a high degree of inter-object occlusion.

5.2 Related Work

Three-dimensional computer graphics has great potential for visualization, but also carries a number of intrinsic problems that threaten its usefulness for information visualization, inter-object occlusion being only one of them. Traditionally, work in this field has been concentrated on navigation and orientation within 3D environments, and many papers exist on this subject.

Some work dealing more directly with object discovery and access in complex 3D environments has also been conducted; the Worlds-in-Miniature technique [132] uses a miniature 3D map of the environment to support both discovery and access, worldlets [53] provide both global overview maps as well as local views optimized for detail, and bird's eye views [58] combine overview and detail

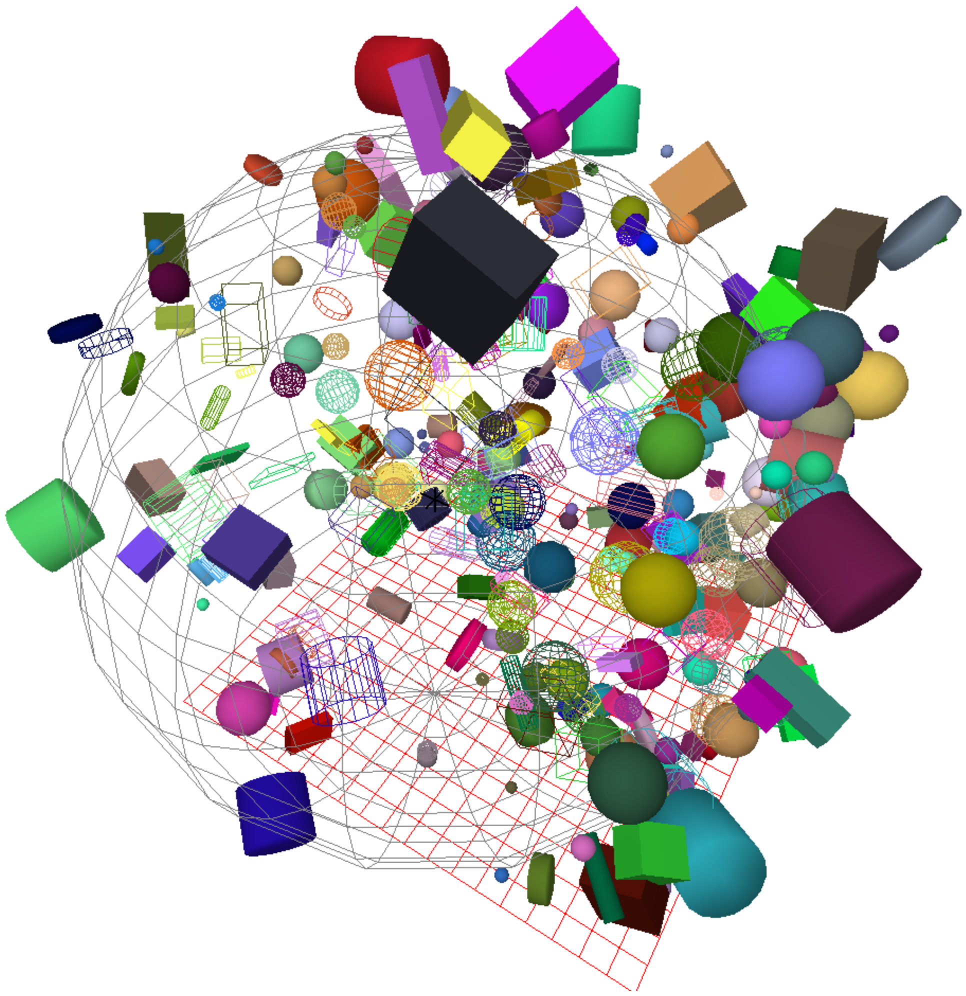


Figure 5.1: The BalloonProbe interaction technique in action on a moderately complex 3D scene.

views of the world.

The BalloonProbe technique disambiguates occluded objects using a form of 3D space distortion, a general approach in information visualization that was introduced by Furnas in his work on fisheye views and focus+context techniques [59]. Other papers where space is distorted to favorably present certain objects over others include the Perspective Wall [100] for visualizing linear information on a 3D surface, the Table Lens [117] for spreadsheet-like tabular visualization, and the Hyperbolic Tree Browser [88] for graphically representing large hierarchies using hyperbolic geometry. In particular, the EdgeLens [145] method bears many similarities to the BalloonProbe, especially its bubble variant, but is intended for selective reduction of edge congestion in 2D graphs.

5.3 The Occlusion Problem

We represent the 3D world U by a Cartesian space $(x, y, z) \in \mathbb{R}^3$. Objects in the set O are volumes within U (i.e. subsets of U). A line segment r is said to intersect the object o if the segment intersects any part of o. A line segment r is **blocked** by an object o if r intersects o. An object o is said to be **occluded** (or fully occluded) from a viewpoint v if all line segments r between v and o are blocked. Analogously, an object o is said to be **visible** from a viewpoint v if there exists a line segment r between v and o such that r is blocked by no object. An object o is said to be **partially occluded** from viewpoint v if o is visible, but there exists a line segment r between v and o such that r is blocked by another object.

Given the above definitions, we can categorize the general occlusion problem as consisting of three main parts:

- *object discovery* – finding all targets $t \in O$ in the environment,
- *object access* – retrieving graphically encoded information associated with each target, and
- *spatial relation* – relating the spatial location and orientation of a target with its context.

Object discovery efficiency is severely hampered by the existence of fully occluded objects, whereas object access also suffers for partially occluded objects. Both issues will affect the efficiency and correctness of users solving tasks using a visualization, but clearly, object discovery is the more serious of the two; if the user is unaware of the existence of an object, she will have no motivation to look for it and access never becomes an issue. The interaction technique presented here will help these subproblems, both by separating occluded objects so that the user can access them, and by making previously unknown objects visible so that they can be discovered. Spatial relation often requires correlating the position of different objects with each other, and thus displacing them might cause problems.

The most common way to overcome partial or total occlusion is to change the position and orientation of the viewpoint. This is akin to the real world, where occluded objects are discovered and accessed by walking around in the 3D environment. In this paper, we instead distort the 3D space as if the user had an inflatable balloon that can push objects away from each other to reduce the occlusion.

5.4 Interactive 3D Space Distortion

The basic idea behind the BalloonProbe interaction technique is to distort the space locally in areas of high object congestion, introducing a spherical "force

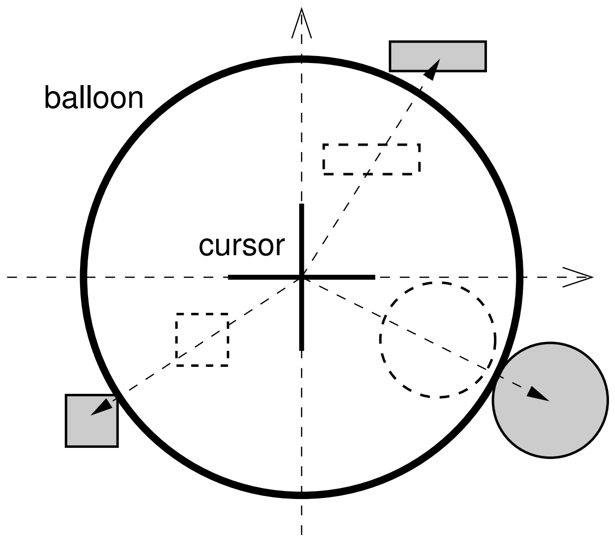


Figure 5.2: 2D overview of the BalloonProbe technique.

field" that repels objects in the visualization to help the user disambiguate between them. This force field is similar to a balloon in that it can be inflated and deflated on the user's command, and the transition from zero to full size and back is performed through a smooth animation. Objects that are in the way of the expanding balloon are pushed away accordingly, and then resume their original position as the balloon is deflated. The effect is that objects are projected onto the surface of the spherical balloon where they can be easily accessed by orbiting the camera around the balloon center point.

See Figure 5.2 for a schematical 2D overview of the balloon distortion (this overview trivially generalizes to 3D). The balloon itself is centered on the 3D cursor controlled by the user, who can not only toggle the inflation/deflation of the balloon, but also modify its full size depending on the situation. This gives the user an intuitive and useful way of quickly separating objects in areas of locally high inter-object occlusion.

5.4.1 Object Displacement

Objects falling within the current balloon volume are simply displaced to its surface along the radius passing from the probe center through the center point of the object. The object is drawn normally in its new position. In addition, a ghosted trace of the displaced object and its displacement vector is rendered, giving the user a visual cue of the effects of the space distortion on the environment and its objects.

To aid the user further in maintaining a cognitive map of the actual layout of the current scene, the balloon itself is visually represented by a wireframe 3D

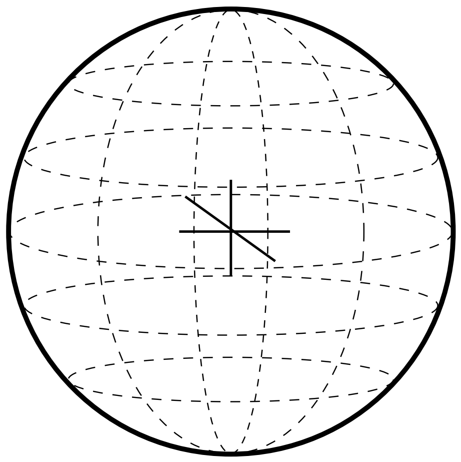


Figure 5.3: Wireframe representation of the BalloonProbe.

sphere, the line segments representing its stacks and slices making it possible to derive the spatial configuration of the displaced objects more easily.

An optional extension to this model is to subject the displaced objects to collision detection so that they do not overlap on the surface of the balloon. The objects are then arranged on the spherical surface using weights proportional to their total displacement distance, conceivably using simple spring equations to derive equilibrium for all objects colliding on the sphere. The result is essentially a force-directed spherical layout. However, this is only necessary for objects contained by other objects (and, in fact, sharing the same center point); otherwise the user can modify the position and size of the BalloonProbe to disambiguate between these.

5.4.2 Selective Displacement

In some cases, it may be useful to be able to *selectively displace* some objects while leaving others in their original position, for instance when the user is looking for a particular type of graphical entity in an information visualization application. This is easily done by configuring the BalloonProbe to not affect a certain type of object, identified by some pattern or identifier. Our prototype implementation (see below for more details) can be configured to selectively displace objects based on their color (either making them fixed and the other objects displaced, or vice versa), but using more complex selection criteria is naturally possible. Figure 5.4 shows an example of this, where all objects except the blue ones have been displaced.

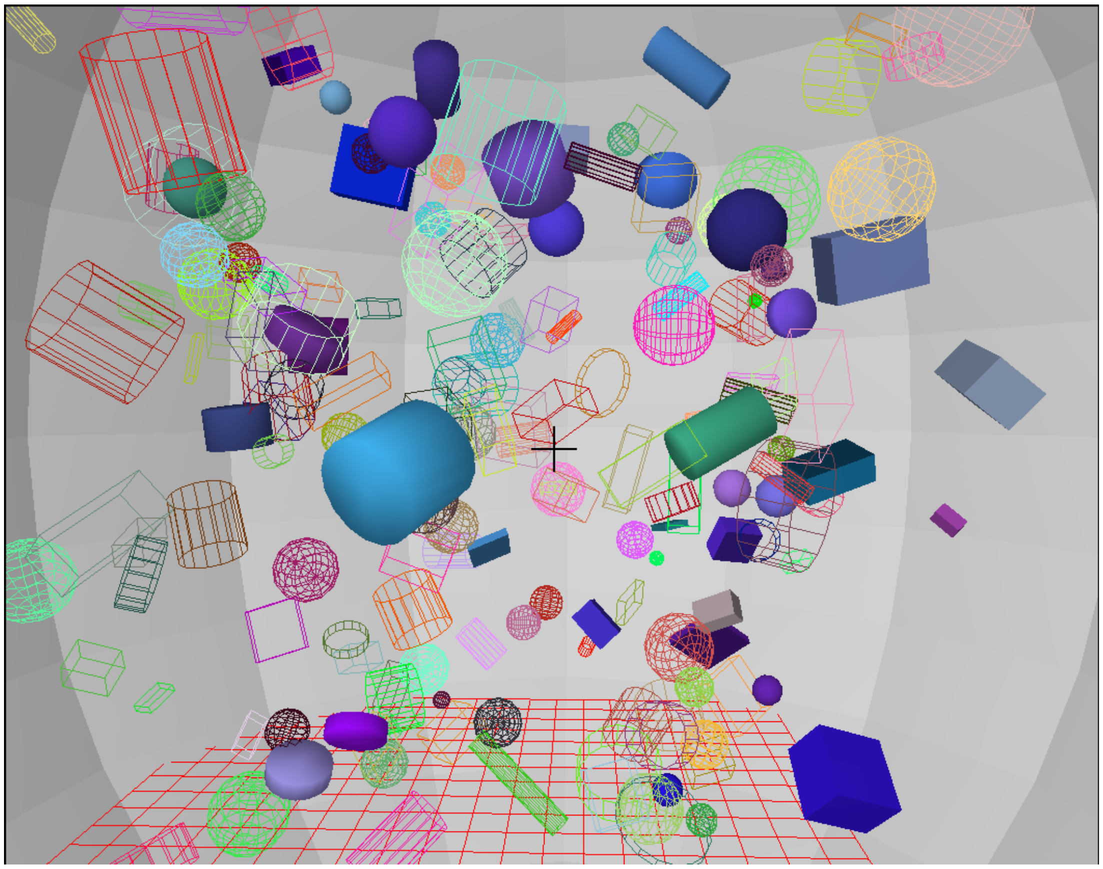


Figure 5.4: Selective displacement of all non-blue objects.

5.4.3 Implementation

We have implemented a prototype implementation of the BalloonProbe interaction technique on a standard Windows-based PC with 3D graphics hardware. The application is written in C# and uses OpenGL for 3D rendering. Mouse and keyboard is used for input, although we have also experimented with other input devices. It is clear that an interaction technique like this would benefit the most from using a high-end 6DOF motion tracker, like an Ascension Flock of Birds system. Our implementation makes use of the Essential Reality P5 Glove instead, which is a low-budget but extremely cost-efficient 3D tracking solution, resulting in slightly worse accuracy and considerably less range than the high-end alternatives.

Even if the display is not immersive, the application renders a simple 3D cursor, allowing the user to control the probe using the input device. The balloon is inflated by pushing and holding a specific button, and deflated when the button is released. Two additional buttons (or a potentiometer) are used to control the size of the balloon. In our implementation, the default inflate toggle is the keyboard control key, and the mouse wheel is used for changing the balloon size.

The test visualization implemented in our prototype is a simple unit cube populated with various random shapes of configurable density. In addition to the BalloonProbe interactions, the application also supports a simple camera orbit operation around the probe center point to allow the user to inspect the displaced objects on the surface of the balloon easily. Figure 5.1 shows an example of this 3D environment with a fully inflated BalloonProbe being controlled by the user to study the graphical objects in an area of high congestion.

5.4.4 User Study

We have conducted an informal user study of the BalloonProbe interaction technique using two volunteers from our research group. The test subjects were asked to experiment independently with the prototype implementation after a short introduction, and were then tasked with solving a few simple visual search tasks. Their comments and feedback were used to improve the prototype. In general, the test subjects indicated that the BalloonProbe was an intuitive and useful tool for quick and effortless occlusion reduction with straightforward and natural interaction.

In the future, we anticipate conducting a formal user study of the BalloonProbe technique using an environment of simple 3D objects (similar to Figure 5.1) where test subjects are asked to perform visual search of specific objects in the environment. By comparing correctness and completion times with and without the use of the BalloonProbe, we hope to show that performance is improved using our new technique.

5.5 Discussion

The BalloonProbe technique addresses occlusion reduction by distorting the display space and wrapping objects onto the surface of a spherical force field. Of course, this is simply a projection from 3D to 2D space, essentially losing one dimension for the visualization. However, the key feature of the method is that the spherical balloon surface is optimized for occlusion-free inspection through simple camera orbiting. The balloon can either be inflated to a size where no overlap occurs, or force-directed collision detection can be imposed on the displaced objects to avoid overlap altogether.

By displacing objects from their original position in the 3D environment, we may lose vital information encoded in the relative positioning of objects in a visualization. This is naturally a weakness of the BalloonProbe, but we hope to allay this somewhat through the rendering of ghosted objects and their displacement vectors. Furthermore, the user is encouraged to alternatively inflate and deflate the balloon multiple times when studying a visualization to note how the individual objects are smoothly displaced.

The BalloonProbe is of special interest in an immersive virtual environment, where the user can use an egocentric viewpoint to actually position herself in the center of the balloon and study it from the inside, getting an excellent view of the displaced objects as well as any objects who were not affected by the selective displacement mechanism of the technique. Thus, in such environments, the probe could be attached to either the body or the manipulator of the user, whereas for non-immersive platforms an exocentric view will work best.

So far, we have described the BalloonProbe as being an invisible or transparent (wireframe) spherical force field that repels objects in the 3D environment. However, for some situations it might be beneficial to render the balloon as solid geometry to further help users separate distracting objects from important ones (see Figure 5.4 for an example).

5.6 Conclusions

We have presented an intuitive space distortion technique for interactively reducing inter-object occlusion in 3D environments. The technique, called BalloonProbe, uses a spherical force field that can be smoothly inflated and deflated just like a balloon, and which will displace any objects it collides with as it expands. By connecting this probe to a 3D-tracked cursor, users can easily pinpoint areas of high congestion and inflate the balloon in this position to separate occluding objects from each other. Ghosted outlines of the objects remain in their original position, allowing the user to maintain a cognitive map of the environment when using the technique. Additionally, users can configure the probe to selectively displace only certain object types while not affecting others, making it easy to quickly clear away irrelevant objects from important objects in a visualization application.

Future work on the BalloonProbe involves testing the interaction technique in a fully immersive virtual environment, as well as trying to integrate it with new or existing VR applications. It would be especially useful to be able to provide an implementation of the BalloonProbe as a standard interaction component for rapid deployment in such applications.

Chapter 6

Evaluating Occlusion Reduction[1]

Niklas Elmqvist[2], Mihail Eduard Tudoreanu[3]

Abstract

We present an empirical usability experiment studying the relative strengths and weaknesses of three different occlusion reduction techniques for discovering and accessing objects in information-rich 3D virtual environments. More specifically, the study compares standard 3D navigation, generalized fisheye techniques using object scaling and transparency, and the BalloonProbe interactive 3D space distortion technique. Subjects are asked to complete a number of different tasks, including counting, pattern recognition, and object relation, in different kinds of environments with various properties. The environments include a free-space abstract 3D environment and a virtual 3D walkthrough application for a simple building floor. Our results confirm the general guideline that each task calls for a specialized interaction—no single technique performed best across all tasks and worlds. The results also indicate a clear trade-off between speed and accuracy: simple navigation was the fastest but also most error-prone technique, whereas spherical BalloonProbe proved the most accurate but required longer completion time, making it suitable for applications where mistakes incur a high cost.

Keywords: occlusion reduction, occlusion management, 3D space distortion, interaction techniques, evaluation

[1]This is an extended version of a paper that appeared in *Proceedings of the ACM Symposium on Virtual Reality Software & Technology 2006*, Limassol, Cyprus.

[2]Department of Computer Science, Chalmers University of Technology, 412 96 Göteborg, Sweden. E-mail: *elm@cs.chalmers.se*

[3]University of Arkansas at Little Rock, AR 72204, Little Rock, USA. E-mail: *metudoreanu@ualr.edu*

6.1 Introduction

Virtual worlds can in general be used for two different purposes: to either mimic reality in an effort to provide understanding about a real place, such as virtual walkthrough applications and photo-realistic rendering, or as a canvas for representing abstract information so that a viewer can make sense of it and reason about it, such as for information visualization. Regardless of purpose, most useful worlds are rich in objects due to the amount of information they have to convey. High object density inevitably leads to clutter and occlusion, causing the virtual world to be difficult to use effectively.

Fortunately, there are many ways of making sense of a crowded 3D world, such as distorting space, using 3D thumbnails, making note of landmarks, utilizing navigational aids, and so on. However, it is often unclear for what kinds of tasks and types of worlds each technique is best suited, i.e. the context of the technique. Examples of such contexts include locating an object in an architectural walkthrough, sifting through volumetric 3D data, or identifying specific shapes in a large collection of objects. Very little work has been done to help designers understand in which situations various techniques work best, especially in an immersive setting.

In this paper, we try to remedy this problem by conducting a comparative study of some popular techniques for occlusion reduction in 3D environments. The aim of this study was primarily to identify situations where different techniques are most efficient in order to help designers make the best choice in terms of efficiency when building their virtual worlds. Subjects were asked to perform a number of different tasks, including basic object counting, relating different types of objects, and recognizing world-sized patterns. We considered different types of worlds that fall into two main categories: abstract 3D spaces populated with 3D primitives, and architectural walkthrough-like environments. For each type of world, we varied the overall object density in order to detect possible points where techniques break down.

Many different object de-cluttering techniques exist in the literature today, ranging from multiple views and space distortion techniques, to those employing transparency or direct manipulation. Out of necessity, this work deals only with a small sample of these. We focus on techniques that are interactive and directly controlled by a user exploring the virtual world and which do not require extra operations such as selection, sorting, and filtering. We also disregard automatic and query-based techniques for eliminating distractors—although such techniques are useful when the objects of interest are already known and selected or grouped together, usage scenarios where the user must explore and determine what they are looking for on the fly are more general and relevant for our purpose.

The techniques included in this study are generalized fisheye views [59], BalloonProbe space distortion [44], and standard 3D camera navigation controls. More specifically, we study two different variants of generalized fisheye views based on object scale and transparency. For the BalloonProbe technique, we study both spherical and wedge-shaped probe geometry. We have built basic implementations of all techniques using a generic test platform, allowing us to conduct experiments of the different methods side by side with the exact same test parameters.

In the next section, we will go through the related work in this field. We then give a general model for the occlusion problem, followed by a discussion of generalized fisheye views and the BalloonProbe as methods for alleviating the problem. The main part of the paper is our description of the user study we conducted and the results we gained from it. We close the paper with our conclusions.

6.2 Related Work

Improving the scalability of virtual worlds with high object density has long been an important issue in the quest to increase the usefulness of 3D environments, and many techniques attacking this problem exist in the literature today. In the following text, we will try to describe the major ones.

6.2.1 Multiple Views

A popular approach to handle object congestion is to introduce additional views that present more information about the 3D environment. This is often done through a combination of overview and detail views. Baldonado et al. [7] present eight general guidelines for designing multiple view visualizations and give examples of existing applications. The Worlds-in-Miniature technique [132] uses an additional miniature 3D map of the environment, allowing the user to discover objects that would otherwise be occluded. The user can also directly interact with the WIM. Worldlets [53] are 3D thumbnails providing both global overview maps of a virtual world as well as local views optimized for detail. They are typically arranged into collections, serving as bookmarks into the 3D world. Yet another multiple-view technique is bird's eye views [58], which combine overhead maps with the standard 3D view of the world.

6.2.2 Space Distortion

Space distortion can be used to manage object congestion in both 2D and 3D, and is typically done by providing one or several foci that serve as the center of attention, and a surrounding context, all integrated into the same view. Generalized fisheye views [59] pioneered and formalized this concept of focus+context views, and many variations on the theme exist. The Perspective Wall [100] uses perspective foreshortening to visualize linear information on a 3D surface, the Table

Lens [117] allows for spreadsheet-like tabular visualization, and the Hyperbolic Tree Browser [88] represents large hierarchies using hyperbolic geometry.

A related approach distorts view space instead of object space; the view can be animated between perspective and parallel projection to facilitate object discovery [50], or multiple viewpoints can combined into one using non-linear projection [1, 124]. Singh and Balakrishnan [125] explore non-linear projection further by introducing fisheye, sticky and mosaic cameras that make use of previous exploration to distort the camera space in response to the user's interests.

6.2.3 Direct Manipulation

Another class of techniques for disambiguating between objects in high-density virtual worlds is invasive in nature, allowing the user to manipulate the objects in the environment directly in order to make sense of it. The EdgeLens [145] is a method intended for selective reduction of edge congestion in 2D graphs and operates by means of a probe-like lens that separates edges that would otherwise overlap each other or even hide graph nodes. The 3D explosion probe presented by Sonnet et al. [130] can be used to separate visual elements in a 3D scene temporarily to create interactive exploding diagrams. In contrast, the Balloon-Probe [44] is closely related to both the EdgeLens and the explosion probe, and provides an inflatable force field controlled by the user that can be used in areas of locally high object congestion.

More complex methods for direct manipulation of objects in virtual worlds exist; representative of these is Selective Dynamic Manipulation [32]. SDM is a suite of 2D and 3D techniques that allow for complex manipulation, comparison and disambiguation. All techniques operate on a currently selected object set; object sets can be freely created, modified and destroyed by selecting individual objects. Visual object properties for a whole set can then be modified by using special object handles attached to each object. This allows a user to, for example, scale up a subset of the objects in a visualization to study their relative sizes without being distracted by objects outside the set.

6.2.4 Transparency

A recent trend in both 3D virtual environments as well as 3D games is to make use of transparency to expose hidden content. Chittaro and Scagnetto [31] investigate the merits of this practice and conclude that see-through surfaces seem to be more efficient than normal 3D navigation, although not as efficient as bird's-eye (overhead) views. Diepstraten et al. introduce view-dependent transparency [42] where occluding surfaces are made semi-transparent to allow hidden objects to shine through. In another work, they instead cut holes in intervening geometry to expose the concealed objects [43]. Coffin and Höllerer [33] present a similar technique with active interaction where the user is controlling a CSG volume that is dynamically subtracted from the surrounding world geometry. A related approach is Viola and Gröller's work on importance-driven volume rendering (IDVR) [140]; here, all 3D elements are assigned a value governing its relative importance, and the final image is a blending of all of the elements with corresponding transparency.

6.3 Preliminaries

As can be seen from the previous section, there is a wealth of available techniques to use for comparison. Given the large scope of the tasks involved, it simply was not possible to select and implement more than a handful of these for our study. Therefore, we had to make a selection.

In order to ensure fair comparison between the techniques, we chose our sample from interactive direct manipulation techniques controlled by a user exploring the virtual world. We only considered general-purpose visual techniques suitable for scenarios when the targets are not previously known—automatic or query-based methods for filtering out distractors or identifying targets are designed for a specific task and do not lend themselves to comparison with a general-purpose technique. Similarly, we disregarded techniques requiring more than one interaction phase, i.e. selecting, filtering or grouping objects or object hierarchies prior to manipulating them. We also consider only single-view methods due to the difficulty of integrating and switching between multiple views in an immersive environment. The interaction techniques required to manage multiple views in immersive environments have not been thoroughly studied, and the effect of these techniques on our measurements cannot be predicted and distinguished from the effects of the occlusion reduction paradigm.

We chose generalized fisheye views and the BalloonProbe technique for the fact that they are both simple and low-level 3D interactions requiring no *a priori* selection or grouping of objects, and they are representative of the space distortion approach to object disambiguation in 3D. Including higher-level techniques such as SDM in the evaluation would certainly be interesting, but the comparison would not be ecologically valid since the scope and usage scenarios of the competing techniques would then be so very different.

In this section, we elaborate on the user tasks targeted and introduce the two occlusion reduction techniques chosen, including the two different variants we have implemented of each technique.

6.3.1 User Tasks

The main user tasks we are targeting with our evaluation are based on the ability to distinguish and identify objects in a given 3D environment. These low-level tasks are always performed in the context of a higher-level task specific to the

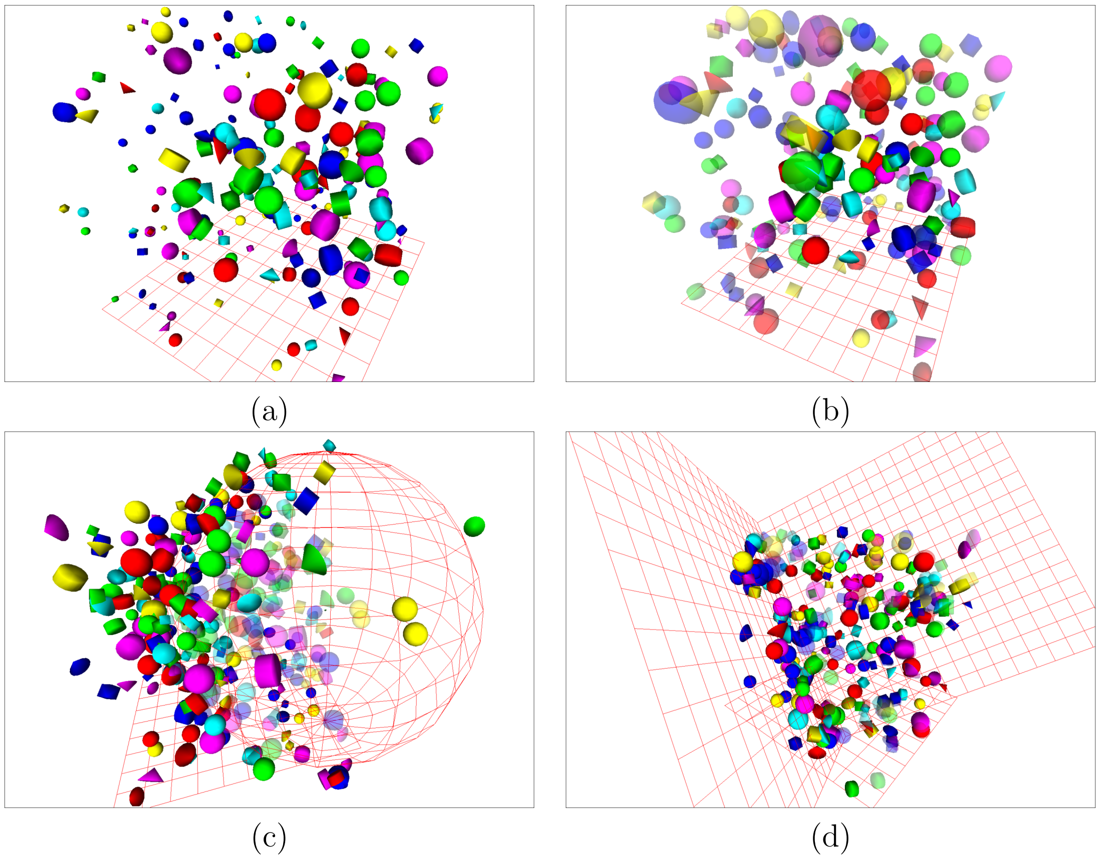


Figure 6.1: Example screenshots of the techniques in the abstract environment. (a) Scale-based fisheye view. (b) Transparency-based fisheye view. (c) Spherical BalloonProbe. (d) Wedge-shaped BalloonProbe.

current visualization. We select a representative set of such high-level tasks in our evaluation in order to give the test subjects a meaningful reference framework.

In this treatment, we refer to objects as being either *targets* or *distractors*, depending on whether they have any relevance to the current high-level task or not. We refer to the low-level tasks as *target discovery*, i.e. the process of finding the targets in a collection of objects, *target access*, i.e. the process of retrieving information in a target, and *spatial relation*, i.e. the commensuration of targets in the world with each other and their context.

6.3.2 Generalized Fisheye Views

Given a general data set to be displayed, a fisheye view consists of a representation of the data centered on a specific focal point in the data set with a *degree-of-interest* (DOI) function governing the *level of detail* of each data point depending on some notion of *distance* between the point and the focus. This is the basic concept of focus+context displays, where the focused detail area of the data set is integrated with the surrounding context in a single view.

The nature of the DOI function controls the level of detail for data points in the fisheye view and is entirely independent of the graphical representation of the data. The function usually depends on the distance between focal and query points and may be continuous, discrete, filtering, or use a semantic scale, etc. The level of detail, on the other hand, is a measure of the information shown in the visual representation.

Implementation

Our implementation of the fisheye view for a virtual 3D environment uses a continuous DOI function based on the Euclidean distance between the viewer's hand position (the focus point) and each object being rendered. We use a standard function with interest inversely proportional to distance, i.e. $i(d) = c/(d + c)$ for a specific constant c.

We developed two different alternatives for the level of detail, one based on object scale and one on object transparency. For the former alternative, objects are scaled according to interest so that more interesting objects are larger than less interesting ones (see Figure 6.1a and Figure 6.3a). For the latter alternative, we analogously modify the overall transparency of an object as a function of the interest.

One interesting point to note is that a 3D environment already has a natural fisheye effect arising from perspective distortion. In other words, objects that are far away appear smaller than objects that are closer to the viewpoint. In some cases, it might even be beneficial to employ a DOI function directly proportional to the distance, causing distant objects to become larger in order to avoid this effect, or nearby objects to become more transparent. This is beyond the scope of this paper, however.

6.3.3 BalloonProbe

The BalloonProbe technique, introduced by Elmqvist [44], provides an inflatable force field probe connected to the user's 3D input device. The probe can be applied to areas of high object congestion in order to disambiguate between objects and reduce the local occlusion. There are two main ways of using the probe:

- **Distractor removal.** Reducing occlusion by giving the users the means to remove distractors from the environment or the view.

- **Target separation.** Reducing occlusion by giving the users the means to separate and isolate targets in the environment.

The task governs the force field geometry that is best suited to solving it. For removing distractors, the displaced objects are not interesting and we just want to get them out of the way, so we use a wedge-shaped force field of two half planes hinged around the probe focus point. The behavior of this probe will be akin to "parting branches" (or distractors) in order to see the targets.

For separating targets, on the other hand, we just want to scatter clustered targets without losing track of them, and instead use a spherical force field like in the original BalloonProbe system. The behavior is then more akin to an actual balloon inflating between targets and pushing them apart to present them for inspection.

Implementation

We implemented both versions of the BalloonProbe, i.e. using the wedge-shaped as well as the spherical probe geometry. The user can inflate and deflate the probe to and from full size using an input toggle button. Another input controls the size of the probe, i.e. the radius of the sphere or the angle between the half-planes. Alternatively, this could be controlled using only two buttons for directly inflating and deflating the probe.

6.4 User Study

We designed the user study with the purpose of identifying the relative strengths and weaknesses of different occlusion reduction techniques. Our intuition was that each technique has a specific context where they perform best. The BalloonProbe technique provides local space distortion and should be effective for tasks with a local scope, whereas fisheye views provide more context and should accordingly be better for more global tasks. Moreover, all techniques should perform better than the base case, the standard 3D flying navigation metaphor with no specific occlusion reduction method. The measures of effectiveness we considered were not only the traditional time and accuracy to perform a task, but also the virtual distance traveled and the number of degrees of rotation required to complete it. The distance and rotation constitute navigational characteristics that are important for designers of immersive worlds because paradigms that require more extensive movement have the potential to cause more fatigue and dizziness to a viewer.

To formalize the above discussion, our hypotheses are:

H1: Technique is a significant effect for time, accuracy, distance, and rotation.

H2: There is interaction between technique and task with respect to our measurements.

H3: Any occlusion reduction technique outperforms simple navigation with no technique on all four metrics.

6.4.1 Subjects

We recruited 16 volunteer participants, four of which were female. The subjects were drawn from a pool undergraduate and graduate students in computer science and engineering at the University of Arkansas at Little Rock. We estimate that participant ages ranged from 20 to 35 years of age. All participants had normal or corrected-to-normal vision, were not color blind, and were able to use the CAVE system and its input devices freely for the duration of the test sessions. They were paid $60 for their effort and competed for an additional prize of $60 for the most accurate result. Three prizes were awarded, one for each of the three tasks in the study.

6.4.2 Equipment

The study was conducted on a three-sided CAVE environment consisting of front, right and floor display walls. The front and right walls are Fakespace reFlex with PixelPipe technology that results in virtually no seam. Each wall is $10' \times 8'$ powered by a Christie DLP projector with a resolution of 1280×1024 pixels. The CAVE is powered by three dual-processor Intel Xeon 3.06 GHz personal computers with 3Dlabs Wildcat4 7210 graphics card, one computer for each display wall. Each display wall provides an active stereoscopic image using CrystalEyes shutterglasses, which are connected to a six-degree of freedom Flock of Birds tracker. Input is provided using a wand with at least three active buttons also tracked by the Flock of Birds. In the early stages of the study, we replaced a defective NeoWand with a Wanda.

6.4.3 Software Platform

We implemented a common test platform for both Virtual Reality as well as standard desktop computers to allow for comparing the various techniques side by side under equal conditions and on potentially different hardware. The platform software is written in C++ with standard OpenGL for 3D rendering. The desktop version uses GLUT, whereas the CAVE version uses the CAVELib SDK from VRCO. Both versions provide a unified framework for implementing tasks, techniques, and scenarios independent of each other using a generic scene graph and extension mechanism.

The software platform supports a simple 3D flying navigation system using the available input devices for each hardware setup—for the CAVE, the view is first-person and controlled by the wand and the 3D-tracked shutterglasses of the user, whereas for the desktop, the view is third-person and controlled by the mouse to pan, orbit, and zoom around the focal point. Note that this study focuses only on immersive environments and, thus, makes use only of the CAVE implementation.

The actual scenario for each trial differs depending on the condition, but all objects (both targets and distractors) are simple 3D primitives, such as spheres, cones, cylinders, and boxes. Each object is drawn in a single color and using standard smooth shading.

6.4.4 Design

The experiment was designed with both between-subject and within-subject variables. Only one independent variable, TASK, had to be between-subject in order to reduce the extended time required by a fully within-subject, factorial design. Henceforth, the tasks are termed "count", "pattern", and "relate". More detailed description of the task, technique, and world are presented in the following subsections.

All other independent variables were within-subject: TECHNIQUE, DENSITY, WORLD, and TRIAL. TECHNIQUE is one of "sphere" for a spherical BalloonProbe, "wedge" for a wedge-shaped BalloonProbe, "scale" for a fisheye based on scale, "transparency" for a fisheye based on transparency, and "no technique" for no occlusion reduction technique. DENSITY, with the levels "low" and "high", referred to the total number of objects populating the scenarios (100 and 200 objects, respectively). WORLD has two possible values, floating 3D objects in space, termed "abstract", and a single-level office environment with walls and doors in addition to the target and distractor objects, termed "architectural". The last variable merely captures the fact that four randomly generated trials were performed for each condition.

The experiment was conducted only in the CAVE system. In the future, we will conduct the same experiment using another group of subjects on a desktop computer, using the "no technique" navigation condition as a cognitive assessment test.

The dependent variables were completion TIME, ERROR, DISTANCE traveled, and total angular ROTATION performed during each trial. The trials were randomized using a fixed pseudo-random seed, ensuring that each subject did the exact same tasks in the same order. Subjects received the TECHNIQUE variable in randomized order to counterbalance systematic effects of practice; for the other variables, the ordering was as specified above.

Tasks

The TASK independent variable represented the task the subject is asked to perform for a specific condition. We created three different types of tasks designed to capture many different aspects of object discovery and access, but all of them involved the lower-level task of recognizing a yellow cone as a target object. The three high-level tasks are presented below:

- **Count.** Count the number of yellow cones that appear in the virtual world.
- **Pattern.** Identify the global pattern formed by the yellow cones in the world.
- **Relate.** Find the third object "spying" on the yellow cone and green box target objects. The targets always appear in triplets in different areas of the world, at a small but variable distance from each other and in different spatial configurations. By observing the objects in the vicinity of the two known target objects, the user is required to find the third target object. The distractors in one neighborhood are different from the distractors in another neighborhood.

The Count task was relatively simple and amounted to the subject merely counting the number of instances of a specific target in a given environment. This task was designed primarily to test discovery in the 3D environment and essentially required no global scope other than forcing the subject to remember which targets had been previously visited.

Relate entailed finding the instances of the two specified targets in the world and then isolating which third object was always present within a certain radius from the two targets (i.e. in a sense "spying" on the two targets). Here, we required a little more correlation between several different sites with the two targets in order to filter out the distractors.

Finally, the Pattern task charged subjects with finding the large-scale shape the target objects together formed in the world; the shapes were the letters C, K, R, X and Y defined as 5×7 grids of objects laid out on the horizontal plane (see Figure 6.2). This task was designed to test global cognition of the 3D environment, and the shapes were chosen so that the individual shapes were easily confused for each other.

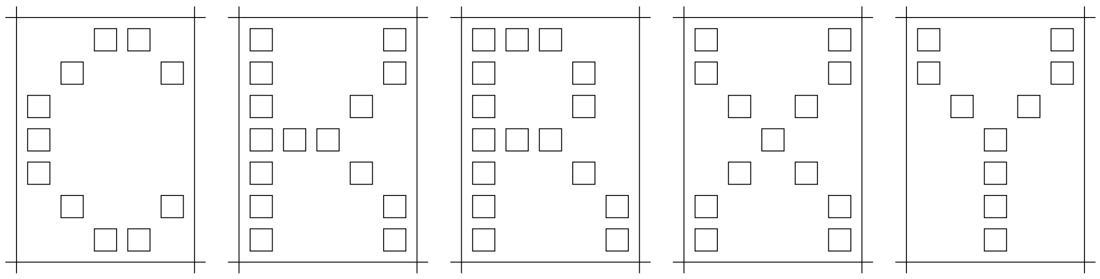


Figure 6.2: The five patterns used in the Pattern task.

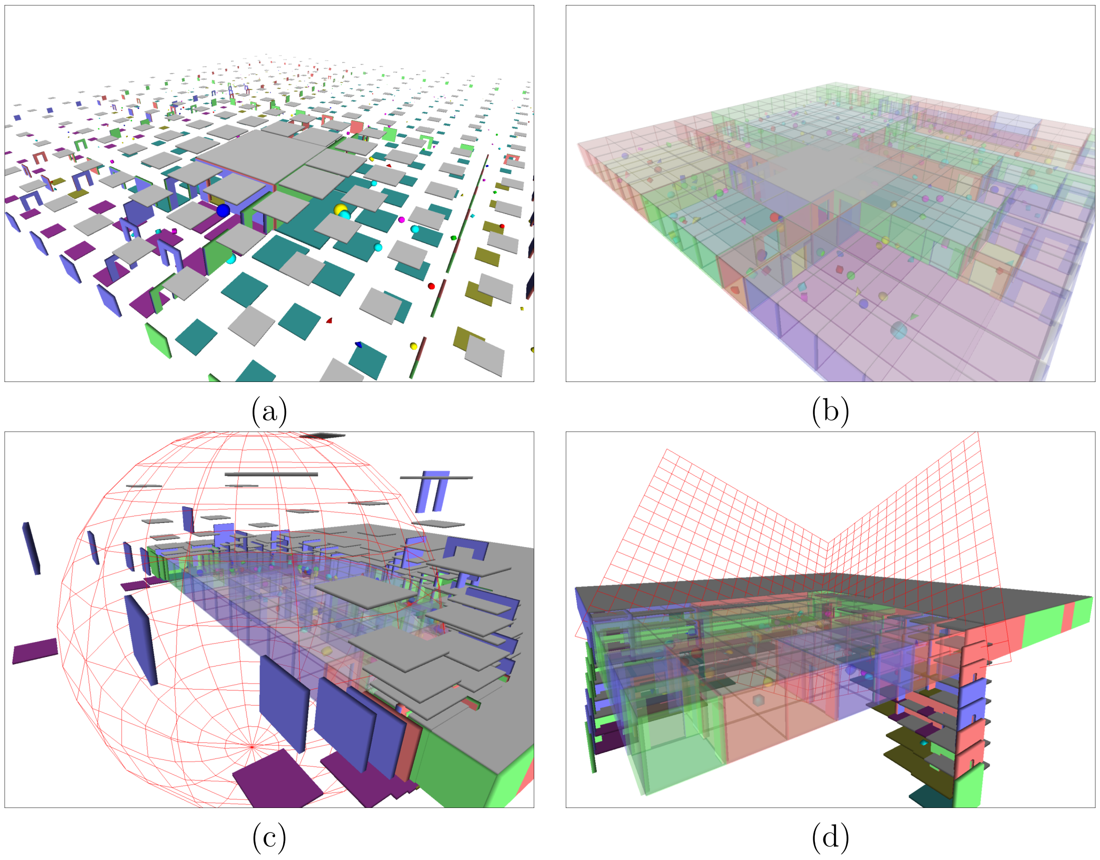


Figure 6.3: Example screenshots of the techniques in the architectural environment. (a) Scale-based fisheye view. (b) Transparency-based fisheye view. (c) Spherical BalloonProbe. (d) Wedge-shaped BalloonProbe.

Techniques

The Technique variable represented the occlusion reduction technique currently in use for a specific trial, and had five different levels: “no technique” for no occlusion reduction (only the 3D camera controls described in Section 6.4.3), “sphere” for a spherical BalloonProbe force field, “wedge” for a wedge-shaped BalloonProbe force field, “scale” for a scale-manipulating fisheye view, and “transparency” for a transparency-based fisheye view. Even if the probe interaction technique typically supports inflating and deflating the force field using a button, this functionality was disabled for purposes of the test and the force field was always active.

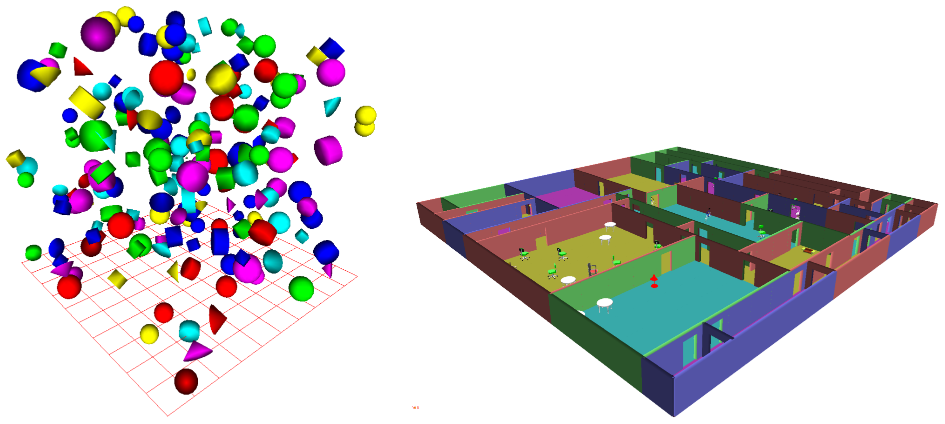


Figure 6.4: Example overview screenshots of the two scenarios implemented in the software platform. (a) Abstract 3D environment. (b) Architectural walkthrough application.

Worlds

The WORLD variable represented the specific type of virtual world for a specific condition, and had two basic levels: "abstract" for a free-space abstract 3D environment, and "architectural" for a virtual 3D walkthrough application. These two types were chosen to represent the two basic classes of virtual worlds that are commonly used in virtual environments; the abstract 3D environment is similar to an information visualization application where abstract data lacking a natural visual mapping is represented by more or less arbitrary 3D geometry, whereas the virtual architectural walkthrough represents the class of virtual worlds that try to mimic reality in some sense. The abstract world only has a basic 3D grid at the bottom of the environment to aid in navigation (see Figure 6.4a for a screenshot). The architectural scenario, on the other hand, provides a basic floor plan of a 3D building with floor, walls, ceiling and doors, all randomized (see Figure 6.4b for a screenshot).

6.4.5 Procedure

The participants were assigned into one of the three tasks (count, pattern, or relate) in a round-robin fashion. A random order in which they were to perform the five techniques was assigned by shuffling of cards; note that to perform the techniques in all possible orders would have required 120 participants. Each technique was to be performed in a separate session. Before starting the first session, however, the participants read and signed a consent form that included a brief description of the experiment. The reading took between five and ten minutes and was kept intentionally short because we considered the interaction techniques intuitive enough to require little explanation.

Five answer sheets were assigned to each participant, one for each session. The sheets were kept in the lab, and participants only had access to the answer sheet while performing the corresponding session.

The first trial was performed under the supervision of a test administrator in order to make sure that the participant understood the task, navigation technique, and trial control mechanism. Participants were reminded that it is their choice whether to sit on a chair provided in the CAVE or stand up and move about the CAVE.

The five sessions, corresponding to the five Techniques, consisted of 16 trials, and the participants were encouraged to schedule them on separate days or at least with a few hours in-between the sessions, because of possible fatigue and dizziness. A few people were able to perform two trials immediately following each other, but never more than two in one day. We conservatively estimate the time required to finish the test at four hours.

A trial began with a blank screen containing a sentence about the task (for a given participant, the task did not change from trial to trial), the buttons required to initiate the trial, and the trial number in order to help participants avoid writing the answer for a trial under the number of another trial. The participants started a trial by pressing a combination of buttons on the wand. The same combination also paused an ongoing trial or resumed a paused one. A paused trial produced a blank screen with a textual reminder on how to resume the trial, but the text was displayed with a different color than the color used between the trials to allow users to recognize that they had not ended the current trial. A different combination of buttons was used to end the trial and move to the beginning of the next one, a blank screen with instructions. Participants were instructed to pause the visualization whenever they needed to ask a question of the test administrator, and to end the task before writing down the answer on the session sheet.

One button was used to increase and another button to decrease the intensity of each technique. The intensity varies with the technique and entails enlarging the spherical BalloonProbe, changing the angle of the wedge probe, or making the objects in the world more transparent or larger for the fisheye techniques. There was no intensity associated with the "no technique" condition.

The software silently recorded, for each trial, completion times, the correct answer based on the randomly generated virtual world, the virtual distance navigated, and the total angular rotation performed by the participant. Trial timing started when the user advanced from the instructions screen and stopped when the subject ended the trial, pausing whenever the trial was paused.

Accuracy of the user answers was determined after the study when the answers written on the session sheets were checked against the correct answer recorded by the computer. The dependent variable Error captured how far from the correct result the participant's answer was in each trial. For the counting task, the error was the absolute difference between the number of target objects present in the scene and the participant's answer. For the pattern task, confusing K and R was considered half as erroneous as any other mistake because the two patterns are quite similar. For the relate task, the answer had two components—the shape of the unknown object and the color of the object—and the error was the sum of mistakes for each component.

The random generator for trials was designed to produce the same scenes in each of the five sessions. Therefore, each TECHNIQUE went through the same 16 worlds (the 16 worlds were different from each other), which allows for a more accurate comparison of the effectiveness of the five techniques. Participants were kept in the dark regarding this feature of the experiment, and they thought all 80 worlds were different from each other. In fact, there were only 16 worlds on which the users operated five times, each time with a different occlusion reduction technique. The answers from a previous session were not available to participants in order to prevent the subjects from recognizing the repeating pattern of trials. Nonetheless, our opinion is that no participant realized that the worlds were repeating. Moreover, there was no difference between the trials experienced by one user and the ones experienced by other users except the order in which techniques were encountered.

Each participant was asked to fill out an informal post-test questionnaire that inquired whether they preferred sitting on the chair or standing up and why, whether they modified the intensity of the technique frequently, and any other thoughts they had about the study.

6.5 Results

In total there were 1280 total trials recorded, but some of the timing data (5 trials), distance, and rotation (77 trials each) were not usable.

6.5.1 Time

The average time spent flying through the virtual environment while performing a task is about 2 minutes and 10 seconds, excluding answer recording and any breaks.

Figure 6.5 depicts the relative timing of the five techniques. Surprisingly, overall "no technique" outperformed all other occlusion reduction paradigms, partly rejecting hypothesis **H3**. However, Figure 6.6 shows that "no technique" was never the best in any of the individual tasks, and its overall performance is because it did not suffer any large penalty in any task. Wedge and transparency were the fastest for both "count" and "pattern" tasks, but they were slower in the "relate" task, where "scale" was the best. Scale in turn had the poorest per-

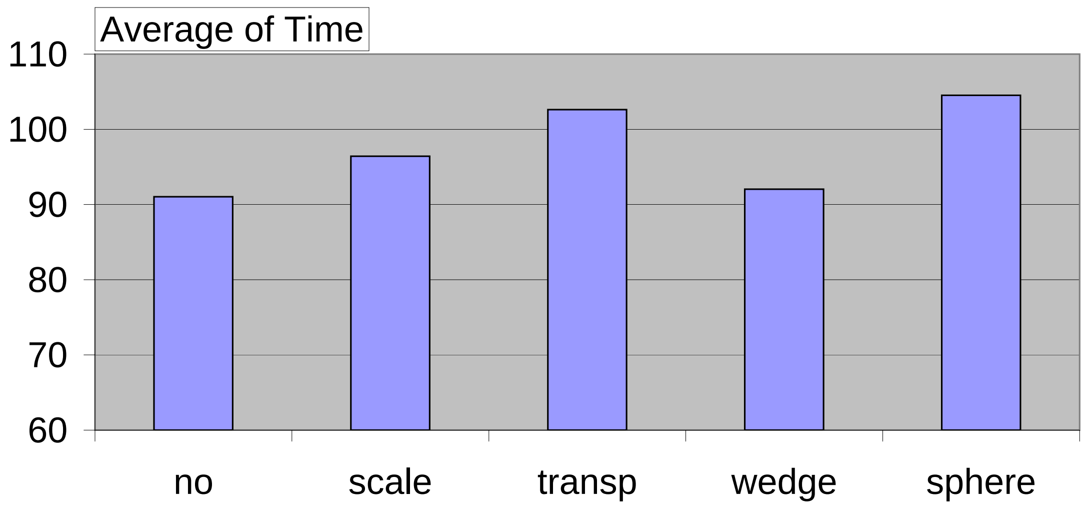


Figure 6.5: Average time spent navigating the virtual world as a function of technique.

formance in “count”. The spherical BalloonProbe has a similar time performance regardless of task.

Statistical analysis of effects shows that Technique is marginally significant ($F(4, 52) = 2.51, p = .0528$), while there is strong interaction between Technique and Task ($F(8, 52) = 4.29, p = .0005$). The technique does not seem to influence completion times differently in different types of Worlds; no interaction was found between technique and world ($F(4, 60) = .67, p = .6178$). The type of the world and density of objects are statistically significant for completion time.

6.5.2 Accuracy

The measure of error is presented graphically in Figure 6.7 and 6.8. Our results show that “no technique” is consistently less accurate than the occlusion reduction techniques. Spherical probe and transparency fisheye support better accuracy.

Technique is a significant effect and there is significant interaction between Technique and Task: ($F(4, 52) = 7.30, p < .0001$) and ($F(8, 52) = 4.50, p = .0003$), respectively. As for completion time, the experiment failed to find interaction between Technique and World for accuracy ($F(4, 60) = .54, p = .7103$). In addition, as for completion time, the world and density are significant factors.

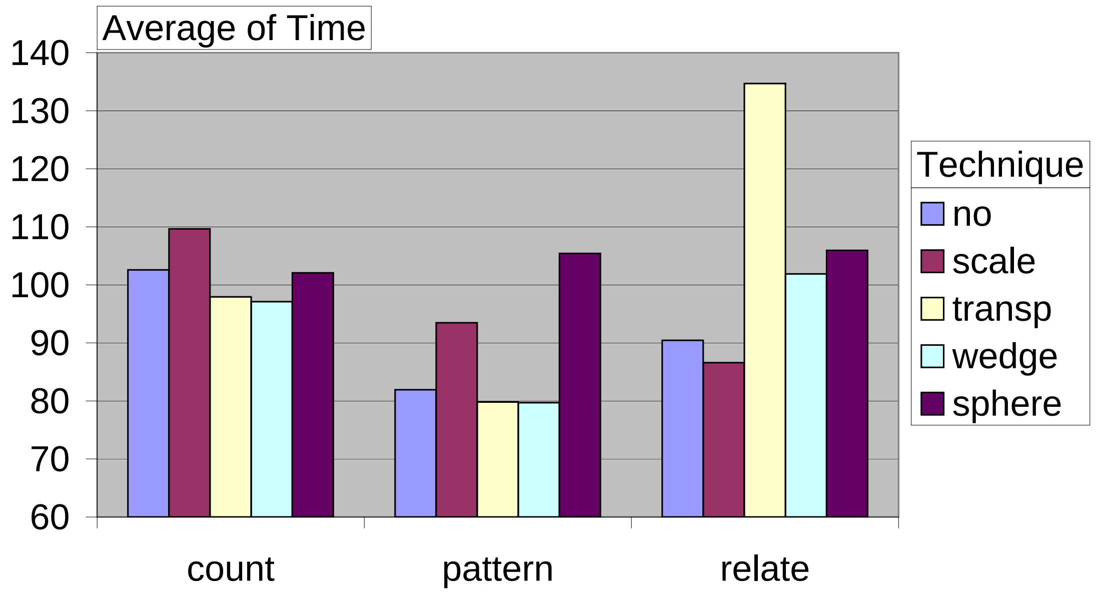


Figure 6.6: Average time spent navigating the virtual world as a function of interaction between task and technique.

6.5.3 Navigation

Navigation was measured along two dimensions, DISTANCE traveled and degrees of ROTATION. The two measurements mirror each other as shown in Figure 6.9 and 6.10. It seems that more flying entails more rotation.

ANOVA shows that TECHNIQUE is a statistically significant factor for navigation ($F(4, 48) = 7.85, p < .0001$ for distance; $F(4, 48) = 6.10, p = .0005$ for rotation). There was strong interaction between TECHNIQUE and TASK ($F(8, 48) = 7.70, p < .0001$ for distance; $F(8, 48) = 5.50, p < .0001$ for rotation). The density of objects failed to register as a significant factor for distance ($F(1, 15) = 2.17, p = .1613$), while both density and world were significant effects for rotation.

6.5.4 Subjective Comments

A number of participants singled out BalloonProbe techniques, both the sphere and the wedge, in their comments about the study and in private conversations with the second author. A simple majority of participants preferred standing up because they felt they had more control over what they saw, and the ones that sat on the chair did so mainly because it was more comfortable. One person wrote that the chair helped with feeling dizzy. Finally, most people said that they frequently changed the intensity of the occlusion reduction technique, especially in the architectural type of world (Figure 6.4 (b)). From discussion with partic-

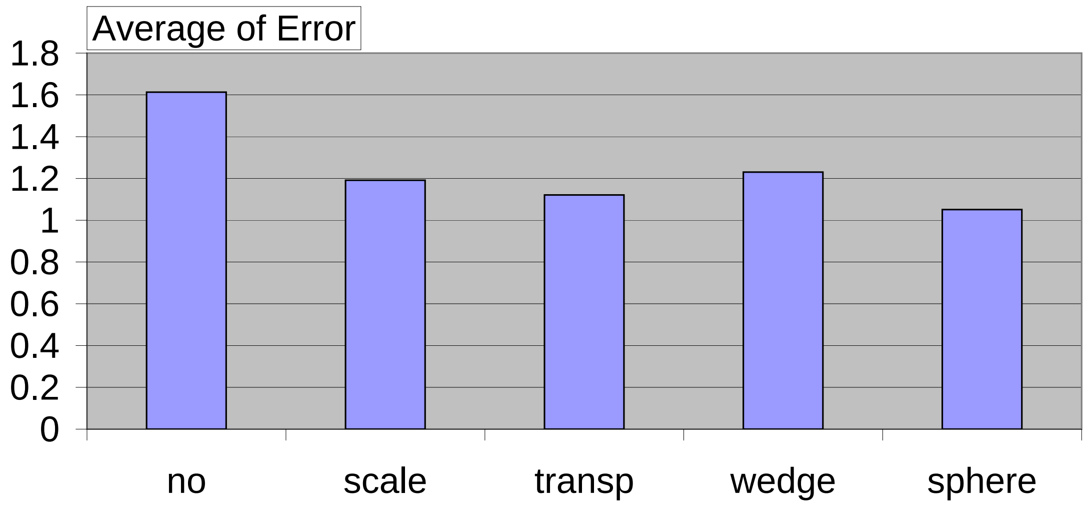


Figure 6.7: Average error per technique (low error equals high accuracy).

ipants, it appears that the architectural worlds were viewed as more difficult by most people, especially for high object density.

6.6 Discussion

The results show a clear trade-off between speed and accuracy. Figures 6.5 and 6.7 show that the fastest technique ("no technique") resulted in the largest error. At the same time, spherical probe, which was among the slowest techniques, proved the most accurate. The implication for the design of interactive, immersive environments is that an occlusion reduction technique is appropriate when time is of no concern, and mistakes incur a high cost, such as in the case of medical applications. For instances when time is important, it appears that simple navigation is more beneficial.

Another design application of these results is that a spherical BalloonProbe may be more appropriate for a wider range of tasks than any other technique, including no technique. That is because, as shown in Figure 6.6, the time required for different tasks incurs little variation in speed and provides consistent accuracy (Figure 6.8). In practice, this is something that a designer values in a general-purpose system: consistency and predictability across various tasks.

Fisheye transparency and wedge probe appear to alter the world in a global fashion because of the amount of navigation performed by the user. Figures 6.9 and 6.10 show that for the one global task, "pattern", transparency and wedge required the user to navigate significantly fewer feet and degrees than any other technique and for any other task. Given this reduced movement, the participants were still able to answer quite accurately and in a short period of time (see Fig-

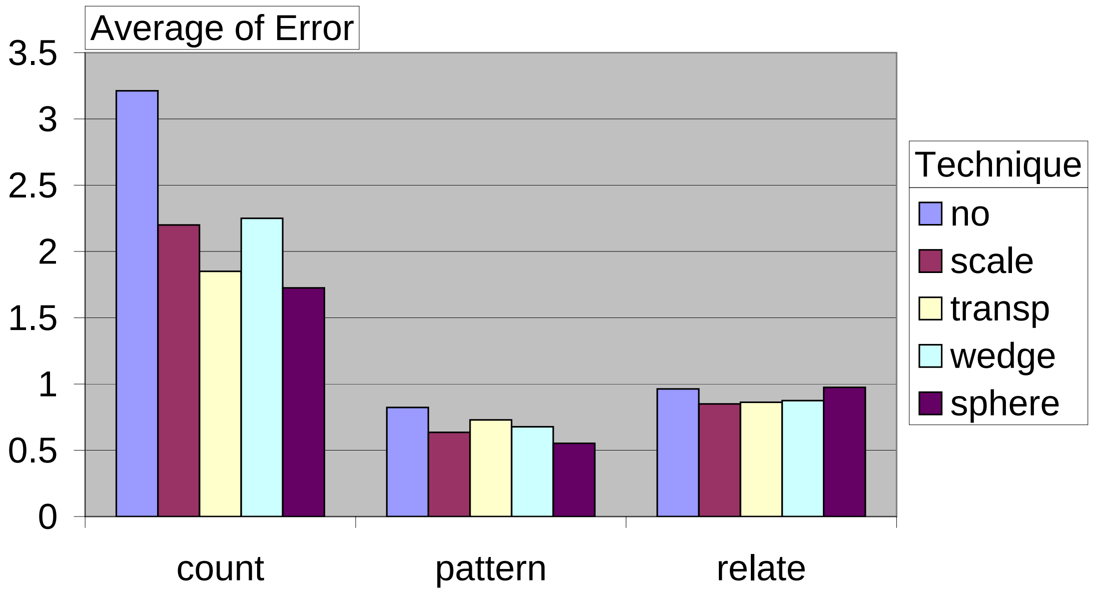


Figure 6.8: Average error per task (low error equals high accuracy).

ure 6.8 under "pattern" for accuracy). Under the pattern task, transparency and wedge had the lowest average time of any technique under any task (Figure 6.6).

6.7 Conclusions

We have presented a comparative user evaluation of two different techniques for managing 3D environments with high object density. The study involved two variants of generalized fisheye views, one using object scale and the other object transparency as the degree-of-interest function, as well as two variants of the BalloonProbe technique, one involving a spherical 3D probe and the other a wedge-shaped 3D probe. Subjects were asked to perform three typical tasks in both abstract 3D environments akin to information visualizations, as well as in a more realistic architectural walkthrough application. We confirmed that the techniques have complementary properties; for example, simple flying navigation is fast, but the spherical probe is more accurate as well as more consistent for a wide range of tasks. The experiment also shows that some techniques exhibit properties that make them desirable for situations where global relationships are important. These discoveries and the characterization of five occlusion reduction techniques are intended to guide designers of immersive, high-object density environments.

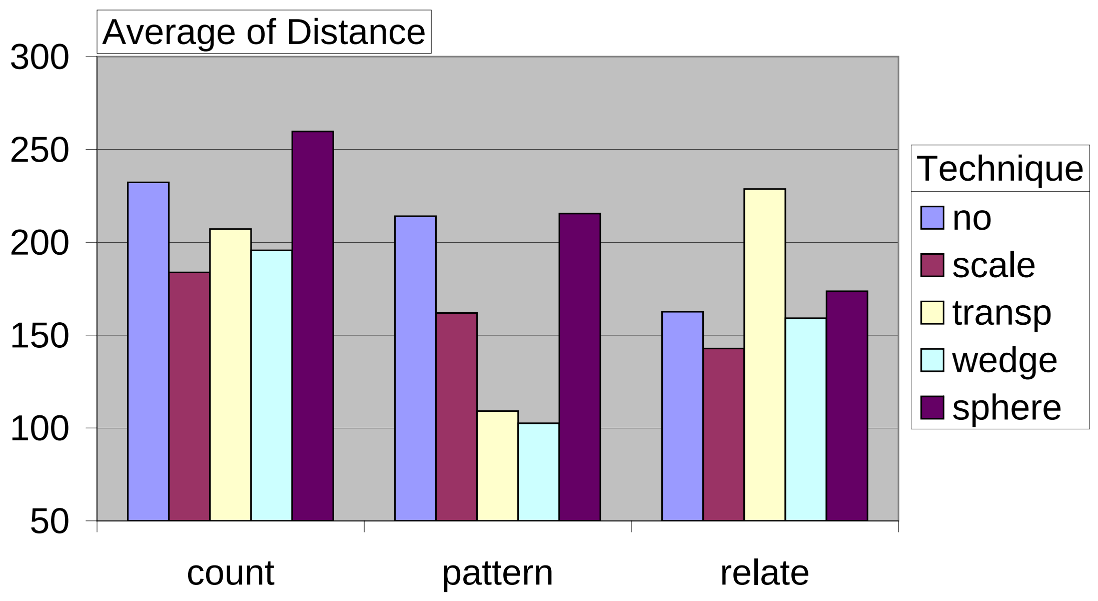


Figure 6.9: Average distance in feet across task and technique.

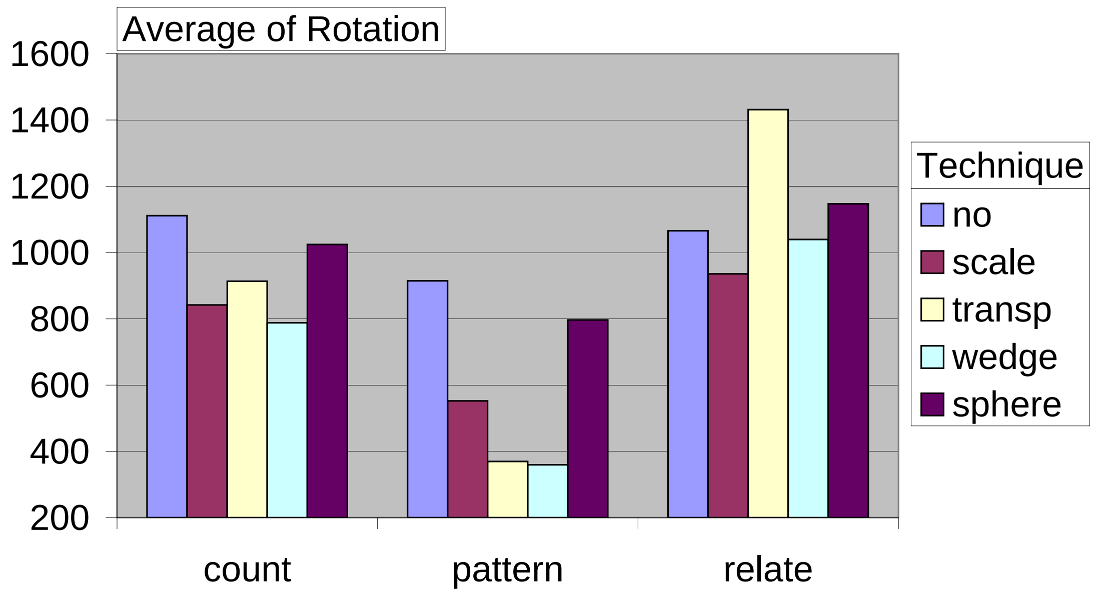


Figure 6.10: Average rotation in degrees across task and technique.

Chapter 7

Image-Space Dynamic Transparency[1]

Niklas Elmqvist[2], Ulf Assarsson[3], Philippas Tsigas[2]

Abstract

We present a method for occlusion management called dynamic transparency intended to support visual perception tasks in complex 3D visualization environments. We also give an image-space algorithm that realizes the method by uncovering hidden targets using per-pixel transparency. The algorithm is based on multiple rendering passes and detects instances of object occlusion in the image-space using the fragment shader capabilities of modern programmable graphics hardware, creating opacity gradients around the occluded objects. We have implemented a prototype version of our algorithm with real-time rendering performance. To evaluate its use, we have also implemented two different example applications portraying different scenarios, including abstract visualization and virtual walkthrough. Results from an empirical user study comparing our technique to standard viewpoint controls indicate that our technique is superior for perceptual tasks in terms of both efficiency and correctness.

Keywords: occlusion management, dynamic transparency, virtual X-Ray, pixel shaders

[1]This is an extended version of technical report CS:2006-10.

[2]Department of Computer Science, Chalmers University of Technology, 412 96 Göteborg, Sweden. E-mail: {*elm,tsigas*}*@cs.chalmers.se*

[3]Department of Computer Engineering, Chalmers University of Technology, 412 96 Göteborg, Sweden. E-mail: *uffe@ce.chalmers.se*

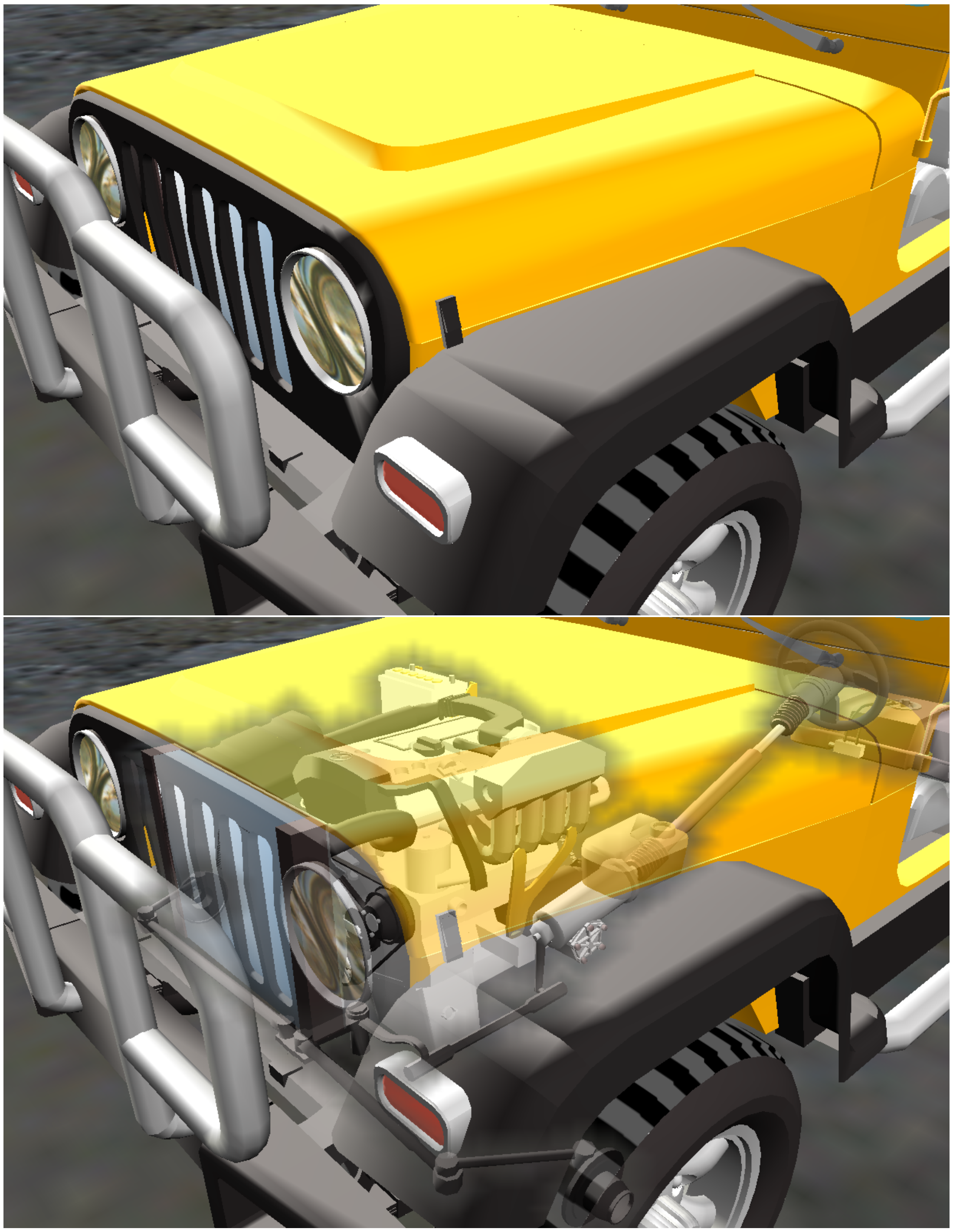


Figure 7.1: Simple 3D scene showing dynamic transparency uncovering an engine inside a jeep.

7.1 Introduction

The ability to utilize the full 3D space as a canvas for information-rich visualization applications is a mixed blessing—while 3D space on the one hand supports an order of magnitude of more layout opportunities for visual elements than 2D space, visualization designers are on the other hand faced with a number of new challenges arising from the nature of 3D space which do not occur in 2D. More specifically, designers must consider the *visibility* of objects when users wish to discover relevant objects, as well as their *legibility* when the user wants to access information encoded in a particular object. For instance, whereas objects that do not intersect can never occlude each other in 2D space, this can very well happen in 3D space depending on the viewpoint and the spatial interaction between the objects. In the real world, this phenomenon is especially evident for a guest entering a crowded cocktail party—the throng of people occludes the individual guests, forcing the newcomer to walk around the whole room to discover his acquaintances as well as to attract their attention. Even if special care is taken in visualization techniques and presentations to avoid these situations, this is very difficult to achieve properly, and is unavoidable in the general case.

In this paper, we introduce a method called *dynamic transparency* to help reduce the impact of inter-object occlusion in 3D environments through transparency, primarily by improving the visibility of important objects to promote discovery, but also by improving legibility to facilitate information access of individual objects. However, this approach may instead introduce additional visual complexity and reduce the user's depth perception. Therefore, our focus in this work lies on evaluating the utility of dynamic transparency for solving visual tasks in both abstract and realistic environments. Furthermore, dynamic transparency cannot be realized using the standard model for transparency, and no real-time performance algorithm exists in the literature that fulfills our requirements. Therefore, we also present an image-space algorithm for dynamic transparency that makes use of fragment shaders for the new generation of programmable graphics hardware to perform occlusion detection in the image space. The effect is somewhat akin to the "X-ray vision" of a superhero.

We have implemented our image-space dynamic transparency algorithm in a C++ application using multiple rendering passes on off-screen texture buffers. We present a number of speedups and optimizations to achieve a good implementation of the technique. Our test results indicate real-time rendering performance on the current generation of programmable graphics hardware on commodity PCs.

In order to evaluate the usefulness of our technique, we have also implemented two different application examples depicting common scenarios within our problem domain: an abstract 3D environment of simple geometric primitives similar to information visualization applications, and a 3D virtual walkthrough application for a complex building environment. Performance measurements on these applications show real-time rendering on standard computer hardware. Furthermore, we have also conducted a formal user study comparing the time and correctness performance of human subjects using our technique as opposed to using standard 3D navigation controls. Results from this study show that the technique allows for significantly improved efficiency for perception tasks compared to standard methods, both in terms of completion times as well as correctness. In general, our dynamic transparency technique seems to be superior for understanding information-rich 3D visualizations, although realism, visual quality and some rendering performance must be sacrificed for this.

The contributions of this paper are the following: (i) a model for dynamic transparency that captures a natural way of achieving high efficiency for perceptual tasks; (ii) an efficient image-space algorithm for dynamic transparency using the new generation of programmable graphics hardware; and (iii) results from a formal user evaluation showing that dynamic transparency significantly improves both time performance and correctness for visual tasks involving discovery, access, and spatial relation of objects in 3D environments.

This paper is organized as follows: We first discuss the related work in the field, including general occlusion management techniques, as well as papers related to employing transparency for object discovery in particular. We then present our theoretical model for dynamic transparency and the occlusion problem in Section 7.3. In Section 7.4, we give our algorithm that realizes the requirements put down in the previous section. Section 7.5 and 7.6 present the user study and our results. We discuss these results in section 7.7. We end the paper with conclusions and a few words on future work.

7.2 Related Work

7.2.1 Navigation and Orientation

No paper on improving the perception of 3D environments is complete without references to the substantial body of work available on 3D navigation and orientation. However, 3D navigation addresses a higher-level issue that is caused by a number of different factors related to 3D environments, occlusion being just one. Nevertheless, relevant work here includes [37, 58, 98] and many more. Bowman et al. define the concept of *information-rich virtual environments* (IRVEs) [18], a combination of information visualizations within the framework of virtual environments, yet many of the challenges [113] presented in their work are directly applicable to any kind of 3D visualization application. The theoretical model for our technique is partly inspired by this work.

7.2.2 Non-Photorealistic Rendering

The approach presented in this paper can be applied to both rendering of abstract 3D information visualizations as well as general 3D models, but is an inherently non-photorealistic rendering (NPR) technique. The emphasis lies on conveying important structural or semantic information about a 3D scene or visualization, not necessarily a high-quality visual appearance.

NPR has become a popular research area in computer graphics in recent years; examples include painterly rendering [76], hatching [115], and edge and silhouette extraction [78]. Although typically not employed directly for visualization, the approach has found use in computer-generated technical illustrations. Gooch et al. [66, 67] describe the use of NPR-based silhouette extraction and tone shading techniques for automatic and interactive technical illustrations. Nienhaus and Döllner [106] present the blueprints method, which employs edge extraction and depth layering to outline and enhance both visible and occluded features of 3D models. Freudenberg et al. [57] introduce another tone-based NPR primitive that may be useful in this context. Our method can similarly be employed for technical illustration, especially with the layer control mechanism discussed in Section 7.3.5, but this is not the primary purpose of our work.

7.2.3 Transparency

The algorithm presented in this work makes heavy use of semi-transparent surfaces to reduce the impact of occlusion in an information-rich 3D environment as well as to avoid the loss of 3D depth cues completely. The general linear model for transparency in computer graphics was introduced by Kay and Greenberg [82]. In order to achieve correct results, transparent surfaces must be rendered in depth order. Everitt [54] discusses the *depth peeling* image-space algorithm for achieving this on modern graphics hardware based on the virtual pixel map concepts introduced by Mammen [101] and the dual depth buffers by Diefenbach [41]. The blueprints [106] technique mentioned earlier uses depth peeling to outline perceptually important geometrical features of complex models using transparency and edge detection. However, depth peeling is a computationally demanding method and interactive frame rates can only be achieved for relatively low depth complexity. Even our simple test scenes can have a depth complexity over 15 counting only front faces. This is currently much too high to be practically solved in real-time using depth peeling, both regarding speed and memory cost, since each layer corresponds to a frame buffer.

Diepstraten et al. introduce view-dependent transparency [42] where NPR transparency techniques are employed for interactive technical illustrations. While closely related to our work in regards to the general method, Diepstraten employs a fixed two-pass depth peeling step to uncover the two foremost layers of transparent surfaces, whereas our method is based on iterative back-to-front rendering and blending, and is thus not limited to a specific depth. Furthermore, where the purpose of our work is to employ dynamic transparency for improved object discovery, view-dependent transparency is primarily aimed at computer-generated technical illustrations.

The use of alpha blending for exposing hidden content in windowing systems is well known (e.g. [72]) but may result in loss of depth cues and legibility. Gutwin et al. [70] explore a dynamically adapting transparency mechanism based on the distance to the mouse cursor to avoid this. Multiblending [8] is a more advanced blending approach where many different image processing techniques are applied separately to different classes of graphical components. Ishak and Feiner [79] take this a step further by introducing a content-aware transparency mechanism that dynamically adapts opacity depending on the importance of various parts of a window. Just like in our work, they employ smooth gradients to emphasize the continuity of the transparent objects. In addition, their system supports a magic lens-like [14] focus filter, similar to the active transparency searchlight in our work (although our version is a three-dimensional magic lens [138]). However, these are all 2D techniques, not 3D like our technique.

Semi-transparency is also commonly used in 3D games and virtual environments to allow users to see through occluding surfaces; Chittaro and Scagnetto [31] investigate this practice and conclude that see-through surfaces seem to be more efficient than normal 3D navigation, although not as efficient as bird's-eye views.

7.2.4 Cut-Away and Break-Away Views

One popular technique for traditional paper-based technical illustrations is called *cut-away* views, where parts of the depicted object is cut away to reveal interior objects that would otherwise be hidden from view. Diepstraten et al. present their work on computer-based cut-away illustrations [43], where a small set of rules are presented to generate an effective model for interactive technical visualization. Cut-away views are not view-dependent, however, and thus bear only superficial resemblance to our work.

In the same paper, however, the authors also present *break-away* views, where interior objects are made visible through the surface of containing objects through image-space holes. While similar to our work, Diepstraten's technique is simplified by semantic knowledge of inside and exterior objects, and the fact that the break-away view is realized by a single hole. To this end, their method is to compute the convex hull of interior objects in a pre-processing step and use it as a clipping volume when rendering. More importantly, their approach does not handle the case when several targets line up and occlude each other, a necessary requirement for dynamic visualizations with a high target density. Our method requires no off-line preprocessing and derives spatial information through sorting and rendering the scene back-to-front, smoothly blending the gradient outline of targets to the scene buffer in an iterative fashion. In particular, our algorithm is designed to handle complex scenes with many targets that potentially occlude each other, always ensuring that the most distant target is visible.

Looser et al. [94] describe a 3D magic lens implementation for Augmented Reality that supports information filtering of a 3D model using the stencil buffer, allowing the user to utilize a looking glass to see through the exterior of a house and into its interior, for instance. This approach relies on the 3D model having semantically differentiated parts, whereas our method requires no such extra information. Coffin and Höllerer [33] present a similar technique with active interaction where the user is controlling a CSG volume that is dynamically subtracted from the surrounding world geometry, again using the stencil buffer. This work does not rely on any semantic target information at all and facilitates exploratory interaction like active dynamic transparency. However, the depth of the volume cutout is limited and user-controlled, and no depth cues from the world geometry are retained other than the cutout border area. With dynamic transparency, as described in this paper, we are guaranteed to always discover occluded objects regardless of depth, and some depth cues are retained using semi-transparency.

7.2.5 Importance-Driven Rendering

A generalization of break-away views, importance-driven rendering assigns importance values to individual objects in a 3D scene and renders a final image that is a composite of not only the geometrical properties of the objects, but also their relative importance. This can be used to achieve various effects for expressing spatial and semantic information about the scene; Viola et al. employ it for importance-driven volume rendering [140] (IDVR) to actively reduce inter-object occlusion in the same way that we do in this work. While the IDVR technique described in the aforementioned paper uses a more general importance scale than the target vs. distractor dichotomy in our model, Viola's implementation (besides being aimed at volume rendering applications) does not provide interactive framerates, whereas our implementation makes use of modern graphics hardware to deliver real-time performance.

In conclusion, none of the previously presented methods fulfills our requirements. Therefore, we present a new virtual X-Ray-based visualization method: image-space dynamic transparency.

7.3 Model for Dynamic Transparency

In this section, we present our model for the occlusion problem in 3D environments and describe the dynamic transparency approach.

7.3.1 Model

We represent the 3D world U by a Cartesian space $(x, y, z) \in \mathbb{R}^3$. Objects in the set O are volumes within U (i.e. subsets of U) represented by boundary surfaces (typically triangles). The user's viewpoint $v = (M, P)$ is represented by the view and projection matrices M and P.

A line segment r is *blocked* by an object o if it intersects any part of o. An object o is said to be *occluded* from a viewpoint v if there exists no line segment r between v and o such that r is not blocked. Analogously, an object o is said to be *visible* from a viewpoint v if there exists a line segment r between v and o such that r is not blocked. An object o is said to be *partially occluded* from viewpoint v if o is visible, but there exists a line segment r between v and o such that r is blocked.

An object can be flagged either as a *target*, an information-carrying entity, or a *distractor*, an object with no intrinsic information value. Importance flags can be dynamically changed. Occluded distractors pose no threat to any analysis tasks performed in the environment, whereas partially or fully occluded targets do, resulting in potentially decreased performance and correctness.

The surfaces defining an object volume have a transparency (alpha) function $\alpha(x) \in [0, 1]$. A line segment r passing through a surface at point p is *not* blocked if $\alpha(p) < 1$ and the cumulative transparency value α_r of the line segment is less than one. Passing through a surface increases the cumulative transparency of the line segment accordingly (multiplicatively or additively, depending on the transparency model).

7.3.2 Visual Tasks

The occlusion problem typically occurs in the following three *visual perception tasks*:

- *object discovery* – finding all targets $t \in O$ in the environment;
- *object access* – retrieving graphically encoded information associated with each target; and
- *spatial relation* – relating the spatial location and orientation of a target with its context.

More concretely, occlusion affects both the visibility and legibility of visual objects in an environment. Fully occluded objects are not visible and cannot be discovered (without manipulating the view or the environment), and consequently are illegible and cannot be accessed. Partially occluded objects are visible and can be discovered (although discovery efficiency is degraded), but their legibility may be low and thus access is difficult. Spatial relation, necessary for many complex interactions and visualizations, requires overview of the whole world, and is thus severely affected by both partially and fully occluded objects.

7.3.3 Dynamic Transparency

The general idea behind dynamic transparency is simple: we can reduce the impact of occlusion by dynamically changing the transparency (alpha) value of individual object surfaces occluding (either partially or fully) a target object. This results in fewer fully occluded objects in the environment and thus directly affects the object discovery visual task.

The fact that the dynamic transparency mechanism operates on the transparency level of individual points of surfaces and not whole objects or even whole surfaces is vital; if whole surfaces or objects had been affected, important depth cues would have been lost. With the current approach, unoccluding parts of a surface will retain full opacity, providing important context to the transparent parts of the object. To give additional context, even occluding surface parts are not made fully transparent, but are set to a threshold alpha value α_T in order to shine through slightly in the final image. There is a tradeoff here: the use of semi-transparent occluders will make object access difficult since intervening surfaces will distort targets behind them. However, it is a necessity in order to maintain the user's context of the environment.

We define our model for dynamic transparency through a number of discrete rules governing the appearance of objects in the world:

(R1) All targets in the world U should be visible from any given viewpoint v.

The first rule is the most basic description of dynamic transparency, and stipulates that no targets should be fully occluded from any viewpoint in the world. Note that a target may still be hidden from the user if it falls outside the current view.

(R2) An occluded object is made visible by changing the transparency level of points $p \in P$ of each occluding surface s from opaque ($\alpha_s(p) = 1$) to transparent ($\alpha_s(p) = \alpha_T$).

The second rule describes the actual mechanics of how to make targets visible through occluding objects. The selection of the set P is not fixed; depending on the application, this could be a convex hull, circle, or ellipse that encloses the occluded object, or the occluded object's actual outline.

(R3) Surfaces can be made *impenetrable* and will never be made transparent.

The third rule provides a useful exception to the initial rule; in some cases, we may want to limit the extent of the dynamic transparency mechanism using impenetrable surfaces (and objects).

(R4) Objects are allowed to self-occlude.

The fourth and final rule provides another refinement of the previous rules; dynamic transparency is performed on object-level, even if transparency management is performed on individual surface points. This means that even if a part of a target is occluded by other parts of itself, none of its surfaces will be made transparent to show this.

7.3.4 Operation Modes

In addition to the basic operation outlined above, dynamic transparency can be used in either active or passive mode. *Passive* mode is the standard dynamic transparency performed on the whole view visible to the user; all occluded objects are revealed automatically without the user having to do anything. This may cause quite a severe impact on the visual quality of the scene, however, and make it difficult for the user to gain an understanding of the layout of the scene.

In *active* mode, on the other hand, the user controls a searchlight (a 2D circle, typically) on the image plane of the scene specifying on which parts of the world dynamic transparency should be active. This is a less obtrusive mode of operation than passive mode and has less impact on the visual quality of the scene, but on the other hand requires direct interaction by the user.

7.3.5 Layer Control

The standard dynamic transparency mechanism, as described above, will peel away all intervening surface layers to reveal occluded targets in a scene. However, in some cases, we may want to control the maximum number of layers to be peeled away. By introducing this capability to the specification of dynamic transparency, we allow for special classes of visualizations, such as the one-layer depth technical illustrations discussed in Diepstraten et al. [42, 43].

7.3.6 Depth Cues

Occlusion is an important depth cue when perceiving a 3D scene, so our approach for dynamic transparency may have an impact on the way users understand the environment. In our implementation of dynamic transparency, we have to ensure that we do not eliminate the occlusion effect entirely, or we will end up with a situation where distant objects occlude nearby objects, so-called *reverse occlusion*.

Fortunately, human perception relies on many more factors besides occlusion to disambiguate depth; examples include stereopsis, motion parallax, relative size, atmospheric perspective, texture gradient, etc. Even if we weaken the occlusion cue, other depth cues will help the viewer to perceive the 3D scene correctly. Nevertheless, this is an effect we want to examine empirically.

7.3.7 Dynamic vs. Standard Transparency

It is important to understand that our requirements, i.e. the four rules presented in Section 7.3.3, can never be fulfilled with standard transparency even if graphics hardware would support correct per-pixel sorting with transparency blending in back-to-front order, which it does not. We cannot simply make distractors covering targets transparent, since we still want objects to self-occlude. That is, objects should still be rendered as solids with only the front-most surfaces visible. Back faces, insides and self-occluded parts of closed surfaces of an object should not be rendered. This cannot be solved with simple back-face and depth culling.

Object-wise, the depth culling should only pass the frontmost surface elements per pixel, and if these elements occlude a target or the close proximity of a target, they should be correctly blended in back-to-front order with a user-specified alpha value in front of the target and fading to no transparency at a specified number of pixels from the target.

7.4 Image-Space Dynamic Transparency

An important observation that follows from our model of occlusion from the previous section is that occlusion can be detected in the image space by simply shooting a ray through the scene for every pixel that is rendered and checking the order it intersects objects in the scene. In modern graphics hardware, this essentially amounts to detecting whenever we are overwriting pixels in the color buffer or discarding pixels due to depth testing. In other words, programmable fragment shaders are perfectly suited for realizing dynamic transparency.

However, correct blending of transparency is order-dependent, and thus our algorithm, as well as most algorithms for transparent objects, requires the objects to be rendered in back-to-front order. This is a classical problem, since current graphics hardware cannot do the sorting for us, although suggestions for solutions exist [24]. Usually, depth sorting is performed on triangle-level. In our algorithm, for non-intersecting objects, it is sufficient to sort on object-level for normal objects that are opaque by default. For intersecting objects, sorting must be performed on a per-triangle-level. Intersecting objects are however rare and usually non-physical. As explained below, objects fully contained within other objects, like objects in a suitcase or nested Russian dolls, can be correctly treated by specifying a fixed sort order between a group of objects.

We divide the scene into groups. By default, a group contains one object. All groups are sorted with respect to their center point, which is precomputed once. The sorting metric is the signed distance to the group from the eye along the view vector. This is better than sorting by only the distance from the eye, because the former corresponds to how the z-buffer works. We use bubble sort, since frame coherency brings the resorting down to an average cost corresponding to $O(n)$.

If some objects are known never to have target objects behind them, like possibly floors, ceilings and outmost walls, those objects can safely be rendered to the frame buffer first.

In certain cases, like for Russian dolls, the sort order between the dolls should be from the innermost to the outermost. A fixed rendering order between the dolls is then user-defined by putting them into the same group with a predefined rendering order, for instance by the order of appearance in the group. In other words, the innermost doll should be rendered first and the outermost doll last. This results in correct transparency, since only the frontmost triangles of the dolls are visible (unlike for classic transparency). If objects are non-solid, like a suitcase or a building, and the inside of the non-solid object should be visible around a target object, then the triangles of the non-solid object should preferably be individually sorted back-to-front. This behavior gives the user a tool to specify which objects that should be regarded as solids and not.

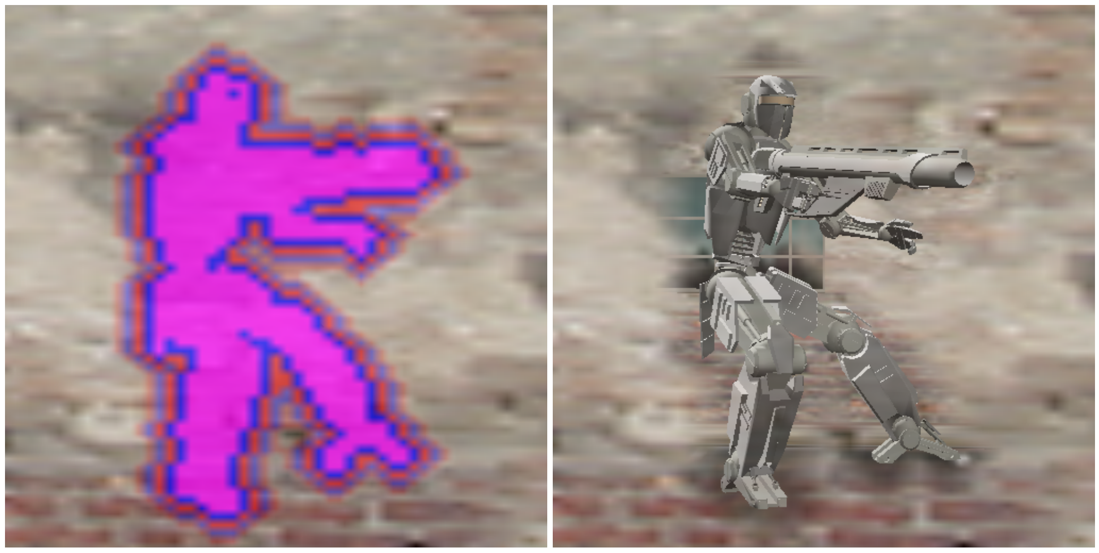


Figure 7.2: Alpha mask creation for an occluded target being made visible by dynamic transparency.

Here is an overview of our algorithm:

1) The groups are rendered back-to-front.

2) All objects are blended into the frame buffer using the value in the alpha-channel of the frame buffer, which defaults to 1 (opaque), as blending factor.

3) Target objects also post-modify the values in the alpha-channel to a value < 1.

The algorithm needs to fulfill these criteria:

- Render all parts of objects (target or distractor) in front of a target object as transparent.
- Render each object as a solid, i.e. only the front-most surfaces should be visible. Thus, the objects cannot be rendered as transparent in an ordinary sense. Back-facing triangles, or more distant front-facing triangles, should not be visible through transparent frontmost triangles.
- Draw a gradual transition from no transparency to a predefined transparency in an n-pixel outline region around each target object (see Figure 7.2).

Algorithm 1 shows an outline of the main algorithm.

Algorithm 1: Main

```
Input: set of groups $G$.
Output: correctly rendered dynamic transparency scene.
1 BubbleSort($G$), taking advantage of frame coherence.
2 for all groups $g \in G$ do
3     for all objects $o \in g$ do
4         if $o$ is a target then
5             renderTargetObject()
6         else
7             renderDistractorObject()
```

Initial requirements for rendering both targets and distractors are that (i) the alpha buffer is initiated to 1 for each pixel at the start of each frame, (ii) rendering is done back-to-front on object level, and (iii) the alpha buffer contains the desired blending factor (transparency) at each pixel. Given these preconditions, we render distractor objects in the following way:

1) Render object to the z-buffer only (using `GL_LESS`), to mask out frontmost surfaces.
2) Blend object to the color buffer (using `GL_EQUAL`).

The first step selects the frontmost surfaces of the object. The second blends these surfaces to the frame buffer, with blending using the alpha values stored in the frame buffer. These alpha values are 1 by default and less in front of, and in an n-pixel region region around, target objects.

In contrast, target objects are rendered in the following way:

1) Render step 1 and 2 as for distractor objects.

2) Render alpha mask, i.e. multiplicatively blend an alpha mask (see Figure 7.2) to the alpha channel of the frame buffer.

The final step ensures that the rendered target is visible by creating a mask that essentially protects the target from being fully overdrawn by subsequently rendered objects.

As specified in Section 7.3.3, the alpha mask can be any type of shape exposing the underlying target, such as an ellipse or circle. We choose the expanded outline of the object with a transparency gradient as the alpha mask shape.

Multiplying a constant alpha value to the pixels covered by the target object is easily done by simply rendering the object to the alpha-channel only and using a color with the alpha value set appropriately. Creating the n-pixel wide surrounding transition is a little bit trickier.

We choose to render to two external off-screen buffers alternately to create a border around the target object with a smooth transition to full opacity. The resolution can be allowed to be quite low. We use a size of 128×128. See Algorithm 2 for pseudo code for the alpha mask algorithm. Refer to Algorithm 3 for the fragment shader pseudo code.

We found that it often looks better to have the transition from full opacity to a low start alpha value α_0 for the gradient outline, while keeping a higher threshold opacity α_T, for fragments directly in front of the target, maximizing both context and discovery.

7.5 User Study

We hypothesize that users employing dynamic transparency for visual perception tasks in 3D environments would be more efficient as well as more correct in performing their tasks than when not having access to the technique. In order to test these hypotheses, we designed a formal user study comparing the new technique to standard 3D camera navigation techniques. We also developed two application scenarios to use in this evaluation.

7.5.1 Subjects

We recruited 16 subjects for this study, three of which were female. The subjects were drawn primarily from our university and were screened to have at least basic computer knowledge. Subject ages ranged from 20 to 35 years of age. All subjects had normal or corrected-to-normal vision, and no participants were color-blind. 12 out of 16 subjects had previous extensive 3D experience.

Algorithm 2: RenderAlphaMask

```
   Input: target object $o$, mask width $n$, two buffers $B_1$ and $B_2$.
   Output: $128 \times 128$ alpha mask blended to the frame buffer.
1  Enable buffer $B_1$.
2  Render the target object $o$ to the alpha channel only, setting the alpha
   values to $\alpha_T$, the threshold transparency for objects in front of target
   objects.
3  Set buffer $B_1$ as texture.
4  Enable rendering to buffer $B_2$.
5  for each layer $\{1 \dots n\}$ of mask do
6      Render buffer-sized quad with the fragment shader specified in
       Algorithm 3.
7      Set the rendered buffer as texture and enable rendering to the other
       buffer. Each iteration adds one pixel-wide layer of the transition.
8      Increase the border alpha value $\alpha_B$ in the shader incrementally starting
       from $\alpha_0$ to 1.0.
9  Disable buffer and activate standard color buffer.
10 Multiplicatively blend the screen-size buffer texture to the color buffer
   (alpha values). Note that resolutions may differ, but linear filtering quite
   efficiently hides zooming artifacts.
11 To avoid ugly jagginess at the pixels along the border of the target object
   due to differences in resolution between the color and mask buffers, render
   the target region again (Line 2).
```

7.5.2 Equipment

The experiment was conducted on an Intel Centrino Duo laptop computer equipped with 2048 MB of memory running the Microsoft Windows XP operating system. The display was a 17-inch widescreen LCD display running at 1920×1200 resolution and powered by an NVidia Geforce 7800 GO graphics card.

7.5.3 Tasks and Scenarios

In order to increase the generality of the study, we designed it to include two widely different scenarios, including an abstract 3D world and a virtual walkthrough in a 3D building, and four different tasks (two per environment). For the former scenario, we designed a basic counting task and a pattern identification task, whereas the latter walkthrough scenario includes both a basic search task as well as a counting task. In this way, we aim to be able to measure not only basic target discovery, but also the more complex visual tasks of access and spatial relation.

In the following two sections, we describe the two scenarios as well as the

Algorithm 3: FragmentShader

```
Input: border alpha $\alpha_B$, frame buffer $F$, screen position $P$.
Output: alpha value $\alpha_P$ for pixel at position $P$.
1 bool IsBorderPixel ← false;
2 for each neighbor $N$ of position $P$ do
3     IsBorderPixel ← $F(N)$.Alpha != 1.0 or IsBorderPixel;
4 IsBorderPixel ← ($F(P)$.Alpha == 1.0) and IsBorderPixel;
5 output IsBorderPixel ? $\alpha_B$ : 1.0;
```

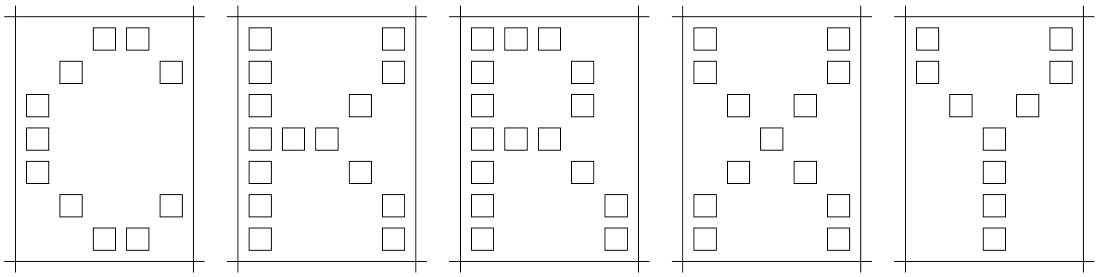


Figure 7.3: The five patterns used in task 2.

tasks associated with each of them. We also give the primary testing purpose of each task in terms of the three visual tasks in Section 7.3.

7.5.4 Scenario: Abstract 3D World

The first scenario (Abstract) is intended to portray an abstract 3D visualization application and consists of a cubic 3D volume of size $100 \times 100 \times 100$ filled with $n = 200$ objects of randomized position and orientation (see Figure 7.6 for a screenshot). The objects are simple unit 3D primitives: spheres, cones, boxes, and torii. Objects are allowed to intersect but not full enclose each other. 10% to 20% of the objects are flagged as targets and the remainder as distractors. Distractor objects are randomly assigned green and blue color component values, while targets were set to a pure red color and made visible using our dynamic transparency technique (for Task 2, distractors could be red as well). The user view is fixed at a specific distance from the center of the environment cube so that no object can fall outside of the view frustum, and can be freely orbited around the focus point to afford view from all directions.

Task 1: count the number of targets (red objects) in the environment. (Purpose: *discovery*)

Task 2: identify the pattern formed by the targets (red cones) in the environment. (Purpose: *relation*)

The pattern is one of the five capital letters C, K, R, X, and Y, rasterized in a 5×7 horizontal grid of the same scale as the environment and rotated in an arbitrary fashion around the vertical axis (see Figure 7.3). The subject is informed of the range of possible letters prior to performing the task, but not the exact rasterizations.

7.5.5 Scenario: Virtual Walkthrough

The second scenario (WALKTHROUGH) is a little more complex in nature and designed to mimic a real 3D walkthrough visualization application more closely. Here, a one-level floor plan is randomly generated from a simple 16×16 grid, creating walls, floors and ceiling as well as ensuring that all rooms were connected with all of its adjacent neighbors through doorways (see Figure 7.4 for an example). A number of $n = 50$ objects are generated and placed in the environment, and all objects are made visible through the walls using dynamic transparency. The 3D objects chosen for this scenario were more complex 3D models, including pets, vehicles, and furniture, yet were easily distinguishable from each other. The user starts each instance in the center of the environment and navigates through it looking for the target using 3D game-like controls involving the mouse and keyboard (mouse to pan the camera around the vertical axis, arrow keys to move, no strafing allowed). The view is constrained to floor level and there is no collision detection with walls or objects.

Task 3: find the unique target in the environment. (Purpose: *discovery*)

Task 4: count the number of targets in the environment. (Purpose: *discovery, relation*)

For the first task, one of the objects in the environment is unique and the user is asked to find this target. The current target is shown in the upper left corner of the screen. After finding the target, the user moves on to mark its estimated location on a 2D floorplan of the environment on a separate screen. See Figure 7.7 for a screenshot.

For the counting task, a random number of the objects in the environment are of the same type and the user is asked to count the occurrences. The current object type is again shown in the upper left corner of the screen. After having estimated that all occurrences are found, the subject enters the amount into the application.

7.5.6 Design

The experiment was designed as a one-way ANOVA for each of the four tasks, with the independent variable DYNTRANS (two levels, "true" or "false"). The

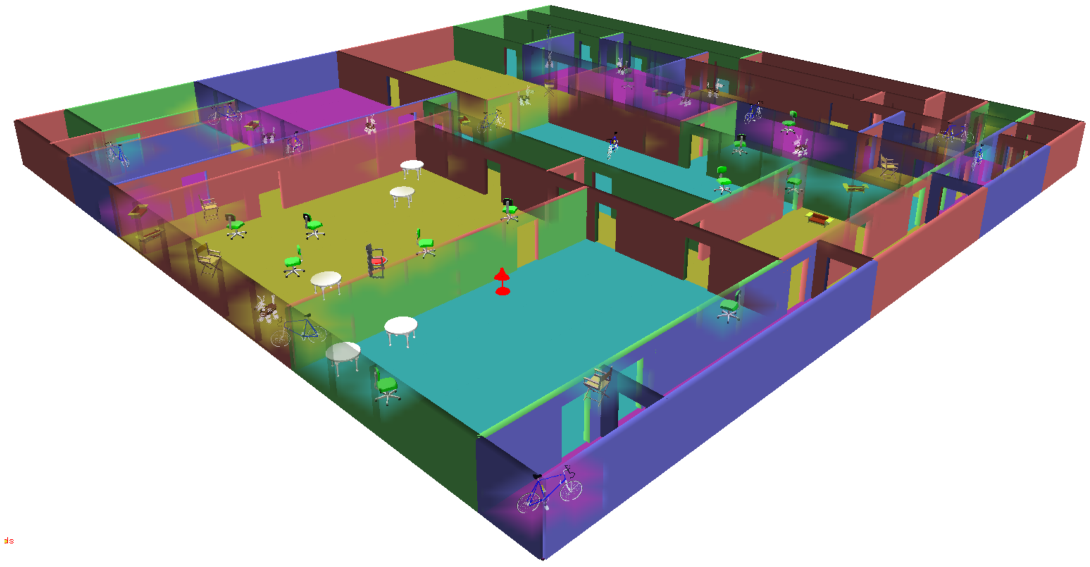


Figure 7.4: Example floorplan for the WALKTHROUGH application with dynamic transparency.

independent variable was within-subjects. The dependent variables included completion times for all tasks, and the error for the counting tasks, error distance for the search task, and correctness for the pattern task. Subjects received both the tasks and dynamic transparency in counterbalanced order to manage systematic effects of practice.

Each task set consisted of three trials per condition. Completion times and user responses to the tasks were collected and silently recorded by the application with no intervention required by the test administrator. Every task set was preceded by a training session lasting up to five minutes where the subject was instructed in the current task and was allowed to explore the scenario as well as ask questions. During the execution of the actual task set, only general questions were allowed.

A full session lasted approximately 45 to 60 minutes. Upon completing all of the task sets in the study, subjects were asked to fill out a post-test questionnaire designed to measure their subjective rating of the new technique as well as identify potential areas of improvement.

7.6 Results

Analysis of the collected measurements indicates that both our hypotheses are correct; subjects are significantly more efficient (i.e. use less time) and more correct when performing visual search tasks using dynamic transparency than

#	Standard	DynTrans	Significance
1	56.26 (38.72)	40.44 (20.99)	$F(1,15) = 7.54, p = .015$
2	22.30 (16.20)	15.80 (10.21)	$F(1,15) = 5.28, p = .036$
3	62.78 (35.63)	23.21 (12.01)	$F(1,15) = 22.98, p < .01$
4	140.0 (61.75)	40.80 (24.16)	$F(1,15) = 48.61, p < .01$

Table 7.1: Average completion times for all four tasks (standard deviation).

without.

7.6.1 Time

Overall, the average completion time with inactive dynamic transparency was 65.17 (s.d. 27.75) seconds, compared to 28.69 (s.d. 11.02) with active dynamic transparency. This was also a significant difference ($F(1,15) = 49.54, p < .001$). Each of the individual tasks also showed significantly shorter average completion times for active dynamic transparency compared to inactive dynamic transparency down to $p < .05$. See Table 7.1 for a summary.

7.6.2 Correctness

For the counting tasks (task 1 and 4), we define correctness in terms of average relative error, i.e. the ratio between the absolute error and the total number of targets for all trials. The absolute error is the absolute difference between the sum of the targets and the sum of the subject answers for the trials. Overall, for task 1 and 4 combined, the average relative error was .100 (s.d. .141) when dynamic transparency was inactive compared to .027 (s.d. .045) when it was active. This is also a significant difference ($F(1,15) = 6.28, p = .024$).

Task 1 in particular showed average relative error of .042 (s.d. .046) for inactive dynamic transparency and .017 (s.d. .018) for active. This too was significant ($F(1,15) = 4.74, p = .046$). Task 4 showed .123 (s.d. .184) and .034 (s.d. .074) average relative error, respectively, not a significant difference ($F(1,15) = 4.12, p = .061$).

For task 2, we define correctness as whether or not the subject identified the pattern as the correct one. This figure was .963 (s.d. .109) for no dynamic transparency and .963 (s.d. .150) for active. This is obviously not a significant difference.

Finally, for task 3, we define correctness as the average Euclidean distance (in world units) from the real position of the target and the point marked on the map by the subject for each trial. With dynamic transparency inactive, this average distance was 16.99 (s.d. 14.44), as opposed to 16.21 (s.d. 8.88). This difference is not significant ($F(1,15) = .068, p = .797$), and indicates that the

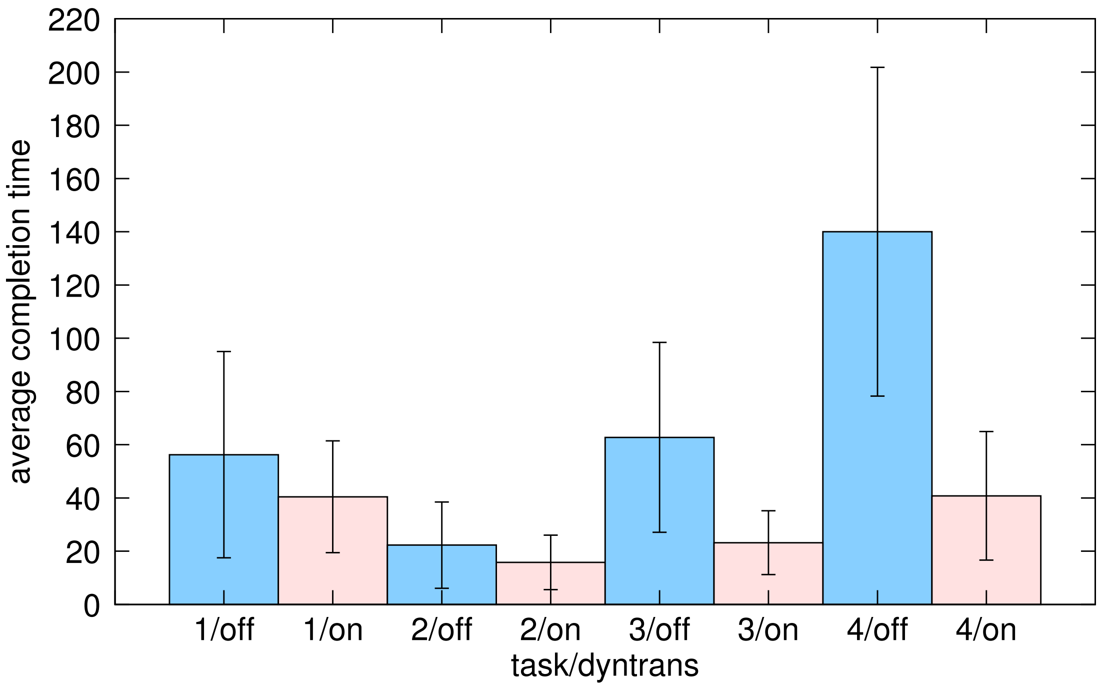


Figure 7.5: Average completion times for all four tasks (error bars show standard deviation).

spatial understanding of the subjects was not negatively affected by the use of dynamic transparency.

7.6.3 Subjective Ratings

Table 7.2 summarizes subjective ratings of dynamic transparency compared to standard vision for the different scenarios. These were all significant differences using Friedman Tests down to $p < .05$. Overall, for all ratings, standard vision had a mean rating of 1.52 (s.d. .59) as opposed to 3.09 (s.d. .38) for dynamic transparency. This is again a significant difference (Friedman Test, $p < .01$).

Overall preference for dynamic transparency as opposed to standard vision was .94 (s.d. .25), with one participant indicating a neutral preference.

Depth perception was quantitatively tested using an example figure from the WALKTHROUGH environment (similar to Figure 7.7), giving an average of 2.94 (s.d. .25) out of 3 correct answers. The average self-reported depth perception on a scale from 0 to 4 was 2.75 (s.d. .45). In other words, subjects felt they still had acceptable depth perception even with dynamic transparency active.

Attribute	**Env**	**Standard**	**DynTrans**
Q1a. Efficiency	A	1.63 (0.72)	2.75 (0.45)
Q1b. “ “	W	0.69 (0.70)	3.63 (0.50)
Q2a. Ease of use	A	1.63 (1.02)	2.63 (0.81)
Q2b. “ “	W	1.38 (1.09)	3.63 (0.50)
Q3a. Enjoyment	A	1.88 (0.96)	2.69 (0.95)
Q3b. “ “	W	1.88 (1.31)	3.25 (0.93)

Table 7.2: Subjective ratings for the two environments (standard deviation).

Application	Resolution	Dyn. Trans.	Framerate
ABSTRACT	800 × 600	no	87
(13,000 triangles)		yes	33
	1280 × 1024	no	87
		yes	33
WALKTHROUGH	800 × 600	no	40
(464,220 triangles)		yes	11
	1280 × 1024	no	40
		yes	11
GAME	800 × 600	no	300
(114,629 triangles)		yes	140
	1280 × 1024	no	188
		yes	90

Table 7.3: Performance for three example applications.

7.6.4 Performance Results

Table 7.3 shows the performance of three example applications with and without dynamic transparency active (the abstract environment, the architectural walkthrough, and the game-like example in Figure 7.1). The test was performed on an Intel Pentium 4 desktop computer with 1 GB of memory running Microsoft Windows XP and equipped with an NVidia Geforce 7800 GTX graphics adapter. As can be seen from the measurements, only the GAME application is fillrate-limited (the bottleneck seems to be buffer switching). For the WALKTHROUGH application, we are performing dynamic transparency on 50 complex objects, so 11 FPS is acceptable, if not quite interactive.

7.7 Discussion

It is important to remember that occlusion is a vital depth cue that humans use to determine the spatial relation of objects in our environment. In essence, the fact that nearby objects occlude more distant ones helps us understand our surroundings. As discussed in Section 7.3.6, the introduction of dynamic transparency may then adversely affect this mechanism, and can actually result in "reverse occlusion", i.e. the phenomenon that distant objects all of a sudden occlude nearby objects instead. In our approach, we address this problem by ensuring that intervening objects made transparent always retain at least some percentage of opacity in order to shine through on uncovered objects. This means that the user receives a visual indication of the existence of the transparent surfaces. Our results seem to indicate that depth perception is still acceptable with dynamic transparency active. Furthermore, in our implementation, we also support toggling dynamic transparency on and off (not during the user study, however); while turned off, the user is free to perceive the 3D environment in the normal way. Active "flashlight"-mode dynamic transparency can also help avoid this problem.

Another factor that is important for context is the shape of the alpha mask uncovering targets. In our image-space algorithm, we make use of an opacity gradient shaped as the outline of the object. Other alternatives would be to use a circle, ellipse, or a 2D convex hull of the object, potentially revealing more of the surrounding context of the target but removing more of the distractors. Our approach strikes a balance between these factors, but the appropriate method depends on the specific application.

Some subjects in our study had the interesting behavior of "respecting" the world more when dynamic transparency was inactive, using the doors in the virtual walkthrough rather than going through walls, whereas they would not hesitate to pass through walls when it was active. While this is an informal observation, this behavior might indicate that the impact that dynamic transparency has on visual realism causes the world to become more ethereal and less believable to the users, thus making them ignore the implicit rules of the environment.

One particular issue with the use of dynamic transparency in particular and transparency in general, is that it often causes a high degree of visual clutter in a scene that would otherwise be far less complex. Many users are simply unused to transparent surfaces and entities, and are easily confused by the side effects of layering and transparency. While this may be a matter of training and habit, it is important to recognize this fact and provide means for the user to turn off the transparency if needed. Alternatively, active transparency, as mentioned above, may be another, less invasive, option.

7.8 Application: 3D Games

The concept of dynamic transparency has applicability beyond the theme of improving object discovery for visualization that this paper focuses on; we believe that it could also prove useful in computer games, where we often may want to suspend graphical realism temporarily by removing intervening objects in order to improve gameplay. Figure 7.8 shows a screenshot from an example application, a would-be 3D strategy game we have developed, rendered in real-time using our algorithm; the player-controlled tank hiding under the cover of the forest is made visible through the foliage in order to help the user see the friendly units.

Figure 7.1 shows another example of a game-like scenario with the different engine components inside a car being exposed to the player using real-time dynamic transparency.

7.9 Conclusions

We have presented a new model for dynamic transparency designed with the purpose of minimizing occlusion of important target objects in 3D visualization applications. This is achieved by dynamically adjusting the transparency of 3D surfaces occluding targets, resulting in "superhero-like" vision yet preserving the important context of the surrounding surfaces and objects. We have further devised an image-space algorithm and implementation realizing this model, utilizing modern programmable graphics hardware to create the desired effect. The algorithm uses the standard framebuffer as a cumulative alpha buffer, rendering the scene back-to-front and blending in alpha masks of target objects to allow for see-through surfaces. In order to verify the technique's usefulness for real visualization applications, we have performed an empirical study evaluating efficiency and correctness gains from using the technique as opposed to standard 3D navigation controls. Our results clearly show that dynamic transparency not only results in more efficient object discovery, but also that users are more correct with the technique than without.

7.10 Future Work

We envision improving our model for dynamic transparency with a more general interest-based scale in the future, allowing users and applications to dynamically specify the relative importance of individual parts of 3D objects to a very high degree (possibly along the lines of the IDVR [140] importance model).

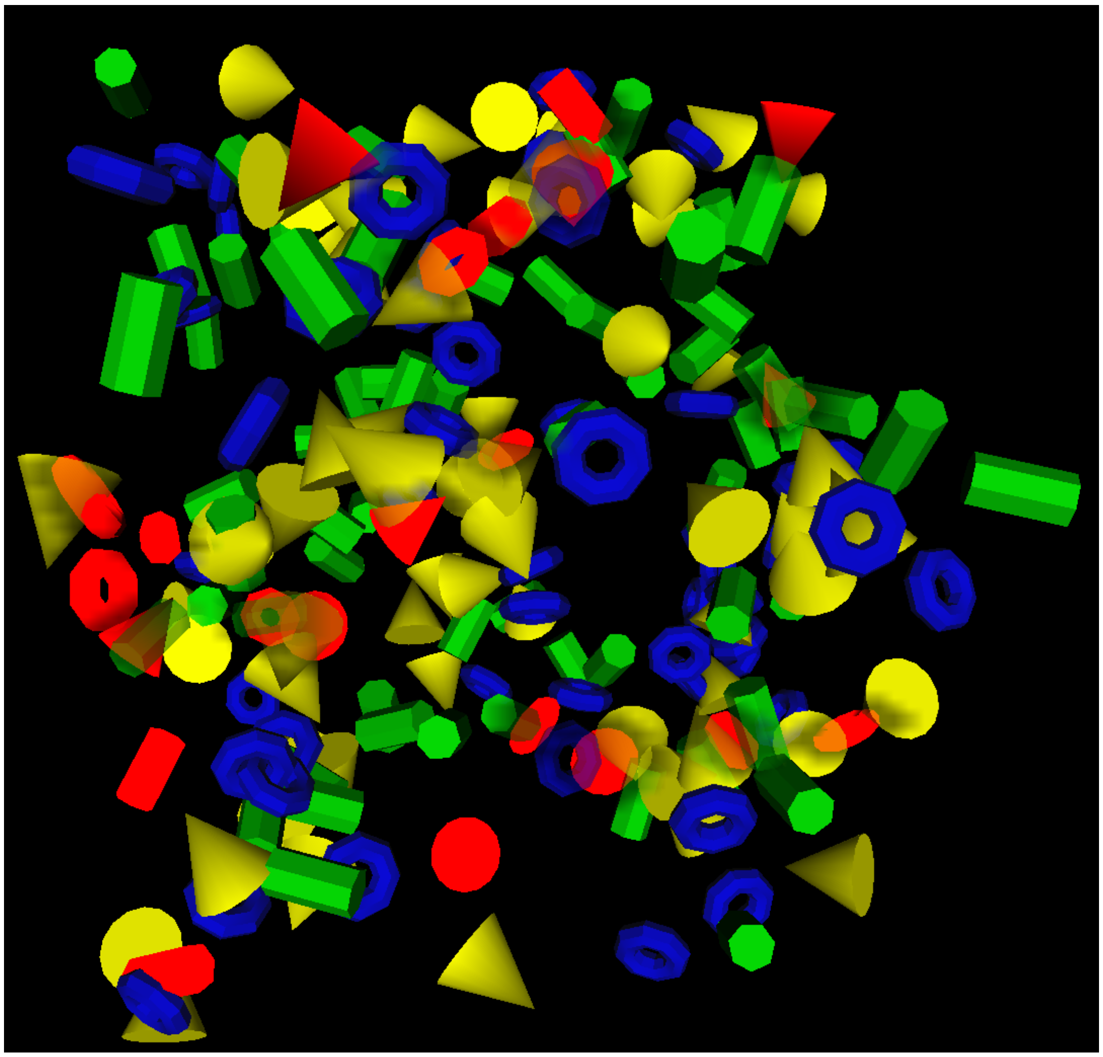


Figure 7.6: ABSTRACT application with dynamic transparency.

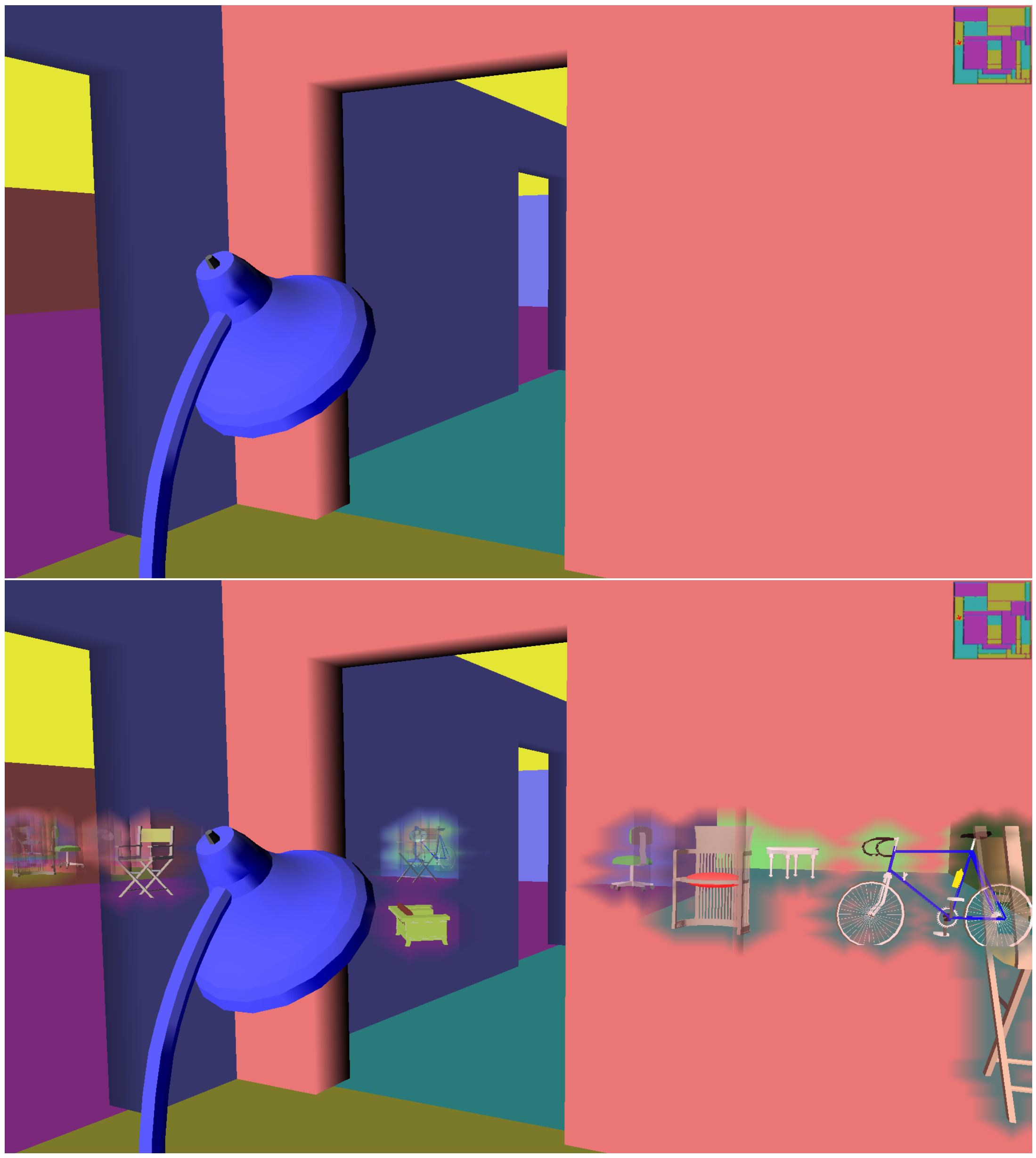


Figure 7.7: First-person view of the WALKTHROUGH application with dynamic transparency inactive and active.

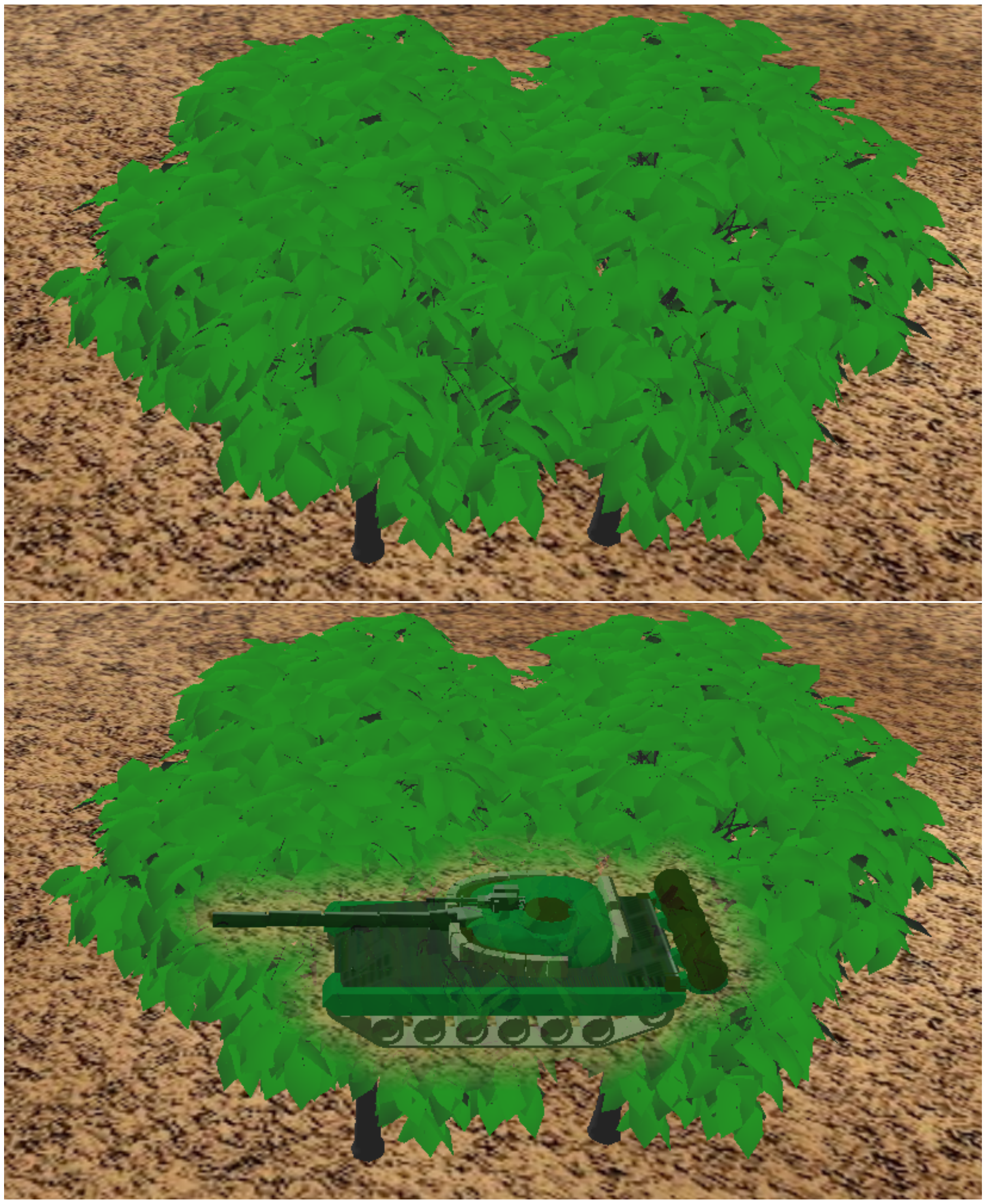


Figure 7.8: Applying the image-based dynamic transparency algorithm to units in a 3D real-time strategy game.

Chapter 8

Navigation Guidance for 3D Exploration[1]

Niklas Elmqvist[2], Philippas Tsigas[2]

Abstract

Navigation in complex and large-scale 3D virtual environments has been shown to be a difficult task, imposing a high cognitive load on the user. In this paper, we present a comprehensive method for assisting users in exploring and understanding such 3D worlds. The method consists of two distinct phases: an off-line computation step deriving a grand tour using the world geometry and any semantic target information as input, and an on-line interactive navigation step providing guided exploration and improved spatial perception for the user. The former phase is based on a voxelized version of the geometrical dataset that is used to compute a connectivity graph for use in a TSP-like formulation of the problem. The latter phase takes the output tour from the off-line step as an input for guiding 3D navigation through the environment using a technique we call spring-zooming. A user study indicates a significant efficiency improvement in performing visual search tasks in a complex 3D environment using the technique in comparison to unaided 3D navigation. Furthermore, the results show that the spring-zooming technique strikes a good balance between guidance and interaction, achieving significantly better general recall performance in comparison to a simple tour-following technique allowing for no user control.

Keywords: tour generation, navigation aid, navigation assistance

[1]This is an extended version of technical report CS:2006-19.

[2]Department of Computer Science, Chalmers University of Technology, 412 96 Göteborg, Sweden. E-mail: {*elm,tsigas*}*@cs.chalmers.se*

8.1 Introduction

Spatial understanding of the structure of a 3D virtual world is vital for a user to be able to navigate and solve tasks efficiently, yet this understanding is exceedingly difficult to attain as the worlds become increasingly complex and increasingly transient [29, 36, 37]. New advances in technology allow designers to increase the visual realism (and thus also the visual complexity) of their 3D worlds to hitherto unseen levels, exacerbating this problem. Furthermore, many worlds are today dynamically created for a specific purpose, such as in response to a search query or as the result of a computation, and will exist only for the duration of the interaction. Our users must then be regarded as tourists in these worlds, lacking specific knowledge about the environment they are exploring, yet in need of solving their tasks as rapidly as possible. There is an obvious conflict in this state of being.

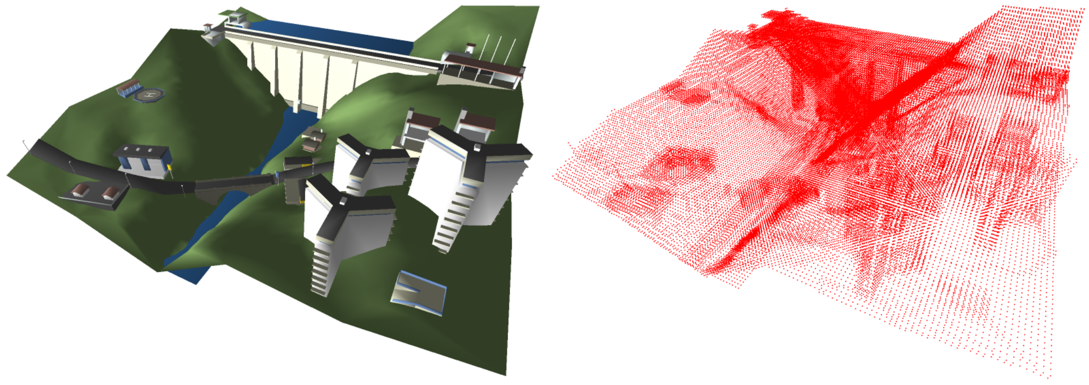


Figure 8.1: Voxelization process for a complex 3D environment.

In this paper, we propose to bridge this gap between the prevalence of complex and unknown 3D worlds, and the user desire to navigate and traverse these worlds effortlessly, using computer-supported navigation guidance. In essence, instead of forcing the user to expend precious time learning a 3D environment, we devise a method to let the computer first explore the environment and extract the vital paths prior to presenting the environment to the user. We then use this information to augment the standard navigation controls, essentially "holding the user's hand" as he or she traverses the world. Depending on the level of interaction desired by the user, we can impose constraints on the path, speed, deviation, and camera direction as the user moves through the world. Furthermore, even if the user wants to navigate freely, the path information can be used to smooth the user's ride, avoid jarring collisions (if collision detection is implemented) or disorienting ghosting through walls (if no collision detection), and ensure that the user visits all targets.

By allowing the user to retain some control over the movement through the 3D space, we aim to make the user an active participant instead of a passive recipient. In order to study this effect, we conducted an empirical user study comparing user performance between varying degrees of control. Interestingly enough, we found no evidence of users having no control performing worse than users having full control. However, subjects who used our hybrid control technique (spring-zooming) performed consistently better than the other two groups.

The main contributions of this paper are the following: (i) an off-line method for automatically computing a tour through a general 3D environment; (ii) an on-line 3D navigation assistance technique based on path data from the off-line step and providing variable interaction; and (iii) an empirical user study evaluating the effectiveness of the new on-line navigation technique in comparison to standard unaided 3D navigation as well as the impact of user control on world recall.

The application domains of this method are many and varied: it can be used for visual storytelling when introducing a new 3D environment, familiarizing a 3D modeler or designer with an unknown or half-forgotten project, presenting all the relevant information in a visualization space, and more. The off-line computation step is designed to be as efficient as possible, providing acceptable tour information with a minimum of time investment. The on-line component can be configured to either be unobtrusive, merely nudging the user in the right direction, or take full control of the user's movement through the world.

This paper is structured as follows: We begin with a review of existing work on 3D navigation in general and navigation assistance in particular. The following two sections describe the off-line tour generation and the on-line 3D navigation assistance techniques, respectively. We describe our user study and the results, and finish the paper with a discussion and conclusions of our findings.

8.2 Related Work

Effective navigation through a three-dimensional computer environment is a well-known problem that has been attacked from many different directions in the past. The problem arises for any environment larger than what can be seen from a single viewpoint, forcing the user to rely on a mental representation of spatial knowledge, often called a *cognitive map* [26, 135]. While navigation can be a challenging task even in the physical world, the absence of many sensorial stimuli in the virtual world compounds the problem even further [29].

Wayfinding is typically defined as a cognitive aspect of navigation with the purpose of planning and forming strategies prior to executing them, i.e. where the actual navigation is not the goal of the interaction but the means to solve some specific task [37]. The wayfinding task is conducted on the user's cognitive map, and thus it is clear that if the user lacks an accurate mental representation of the environment, performance will suffer.

8.2.1 Spatial Design

One approach to improve wayfinding is to organize the virtual environment in a way that promotes understanding and orientation, in essence making it easier for the user to construct an accurate cognitive map. Due to the similarities with navigation in physical space [139], we can leverage existing research from urban planning, geography, and psychology. For example, Darken and Sibert suggest a number of design guidelines for organizing a virtual environment to facilitate the acquisition of spatial knowledge [37], further extended in [36]. Similarly, Vinson [139] argues for the importance of landmarks for navigation in a 3D world, and gives a comprehensive set of guidelines for their placement, design, and composition.

8.2.2 Navigation Widgets

Visual aids can be used to great effect for improving 3D navigation. Chittaro and Burigat [29] present an array of different compass-like navigation widgets for helping the user to find important objects and places in a virtual environment. Trails [121] help users utilize previous explorations to improve their current search. Path drawing [77] lets the user draw an intended path directly on the 2D view of the world to aid navigation.

8.2.3 Motion Control

Another powerful class of navigational aids is motion control, i.e. different methods of traveling through a virtual environment and potentially guiding or constraining the user's movement. Techniques in this class can have varying degrees of obtrusiveness, from merely nudging the user in the right direction to constraining or downright controlling the viewpoint completely. Bowman et al. [17] present a taxonomy of first-person motion control techniques for manual viewpoint travel that is useful for evaluating such methods.

The flying, eyeball-in-hand, and scene-in-hand metaphors [141, 143] constitute perhaps the most basic motion control techniques with practically no automatic control. Mackinlay et al. [98] describe a method of logarithmically controlling the viewpoint speed while moving through a 3D world to allow for rapid motion over large distances, yet slowing down when approaching the target. In related work, Song and Norman [129] propose a set of non-linear motion control techniques for intuitively traversing virtual environments. The work of Tan et al. [134] on a moded navigation technique is interesting, not only for the fact that it contextually combines two different motion control techniques (flying and orbiting), but also that it couples the speed of movement to the height and tilt of the camera to smoothly support both local detail views and global overviews.

Guided navigation techniques exhibit a little more control on the motion of the viewpoint, allowing the computer to augment the user's spatial knowledge with additional information. Wernert and Hanson [144] present a taxonomy of assisted navigation, and also discuss a "dog-on-a-leash" approach to guidance through a 3D world. This approach is similar to the "river analogy" introduced by Galyean [61], where the viewpoint is tethered to a vehicle following a path through the virtual environment and some degree of control is retained by the user. The motion control technique presented in this paper is similar to both the river and dog metaphors, yet supports variable interaction to a higher degree. Furthermore, unlike these two papers, our work also includes an empirical user study comparing both the effectiveness of guided navigation over unguided navigation, as well as the impact of user control on world recall.

Another notable technique is the virtual guide of Chittaro et al. [30] which the user must follow actively; the guide's path is also automatically computed using an algorithm operating on a 2D occupancy matrix similar to the tour generation algorithm in this paper, but our method can handle any general 3D environment and not just one-floor buildings.

Finally, constrained navigation techniques essentially assume full control of viewpoint motion, sometimes even moving the gaze of the user in the desired direction. By reducing the freedom of the user, navigation and wayfinding can be simplified, and eliminate the need for expensive features such as collision detection. Examples of this approach include that of Hanson and Wernert [71], who employ invisible surfaces to constrain user movement, and of Andújar et al. [4], whose Way-finder system algorithmically computes an exploration path through a 3D environment. The latter algorithm is based on a voxelized version of the 3D world, just like the tour generation algorithm presented in this paper, but uses another method to compute a cell and portal graph for use with a backtracking tour generator, whereas our algorithm builds disjoint visibility subsets and performs TSP computations on the resulting connectivity graph.

8.3 Overview

Historically, the term "grand tour" used to refer to the peculiar rite of passage that young European (typically British) noblemen undertook as more or less part of their education during the late 1600s until the 1800s. The tour was essentially a travel itinerary of Europe, designed to expose the neophyte to as many of the important cultural and historical landmarks as possible [60]. In scientific visualization, on the other hand, a grand tour is a method for viewing multidimensional data using orthogonal projections onto a sequence of lower-dimensional subspaces [5].

In the context of this paper, the term is perhaps more literally related to the historical use of the term than the mathematical method. The basic idea is to take the user on a sightseeing tour of a certain 3D world in order to help him or her in

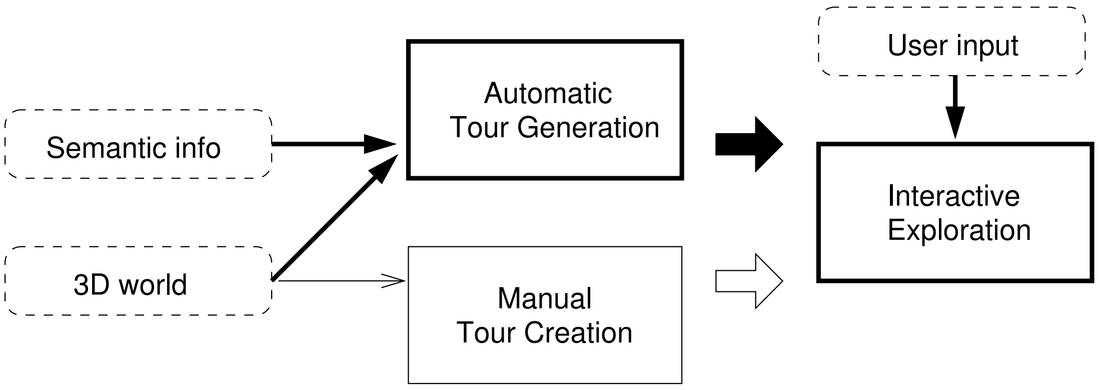


Figure 8.2: Navigation guidance overview.

understanding its structure and important landmarks. For this purpose, the tour has been designed so that it visits all of the landmarks in some suitable order. It either can be built manually by a human designer, or computed automatically using a tour generation algorithm (the approach taken in this paper). The tour is then used for guiding the user in interactively exploring the world.

See Figure 8.2 for an overview of the two-step process presented in this paper. The following sections will give the details on these two phases.

8.4 Automatic Tour Generation

The objective of the automatic tour generation phase is to build a grand tour of a 3D world given the following input:

- a 3D world geometry dataset;
- a set of landmarks; and
- a starting point.

The tour should start and end in the starting point and visit all of the landmarks in the world. Beyond these simple requirements, we can add a few more: the generated tour should be "good" in some sense, and the process should be robust in the presence of inaccessible landmarks, i.e. landmarks that are landlocked and cannot be visited due to surrounding geometry.

The definition of a "good" tour is open to debate; in this work, we take it to mean a tour of as short length as possible (not necessarily optimal) that visits all landmarks as few times as possible. Furthermore, the tour should not stray outside the bounding box of the 3D world to avoid trivial (but impractical) solutions where the viewpoint is placed at infinity.

An important observation is that visiting a landmark in this context is equivalent to seeing it, so it is not necessary (and in fact undesirable) to pass through the same spatial location as the landmark for it to be regarded as having been visited. At the same time, our algorithm allows for specifying a maximum visibility distance, i.e. the furthest away the tour may pass a landmark in order to visit it.

Finally, the representation of landmarks is significant; in our implementation, we choose to use 3D points for simplicity, but the algorithm can support other 3D primitives as well as actual 3D triangle meshes as targets.

Figure 8.3 gives a rough outline of the sequential tour generation process. We explain each of these steps in more detail in the following subsections.

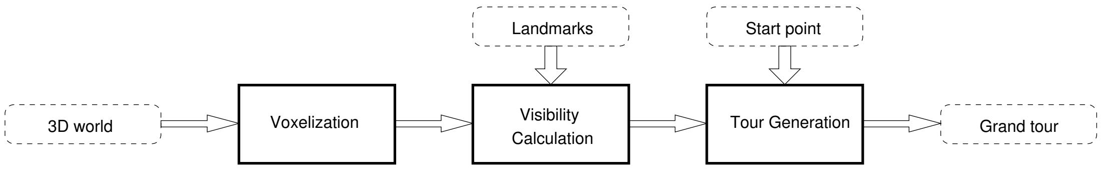


Figure 8.3: The tour generation process.

8.4.1 Voxelization

Our tour generation algorithm operates on a voxelized version of the 3D world, so the initial step of the process is to voxelize the geometry dataset into a volume representation (see Figure 8.1 for an example). We first compute the bounding box of the world and enlarge it in all directions by a single voxel width to allow for the algorithm to skirt along the perimeter of the 3D world if necessary. Then we voxelize the world using incremental 3D scan-conversion.

The process of incremental 3D scan-conversion builds a volume representation of a 3D boundary representation such as a triangle mesh by iteratively scan-converting the 3D primitives into a voxel buffer. Kaufman and Shimony [81] give algorithms for scan-converting all manners of 3D primitives; our method is based on a recursive subdivision of 3D space into an octree representation and testing the triangle against each volume using a fast triangle-box intersection test [2] (see Figure 8.4).

8.4.2 Visibility Calculation

Armed with a volume representation of the 3D dataset, we can now calculate the visibility information of all voxels given the set of landmarks the tour should visit. We use an integer one-pass voxel traversal algorithm [93] to determine if there is a clear line of sight between the current voxel and a specific landmark

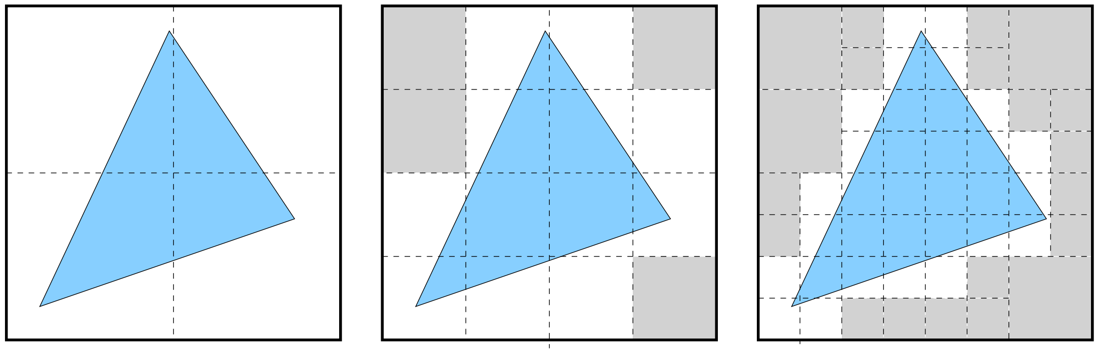


Figure 8.4: Recursive 3D scan-conversion using an octree.

(this particular step must be generalized for landmarks represented as something more complex than a point). Additional constraints can also be imposed at this point; we currently ensure that the distance between the voxel and the landmark is within the maximum visibility distance, but other constraints are plausible.

Having derived the visibility information for all voxels, we then group them into disjoint subsets that we call *visibility sets* using a breadth-first search algorithm. Each set is built so that its members are contiguous and have the same visible landmarks. A special case is made for voxels with no visible landmarks; they form "zero-visibility" sets, and are necessary for connectivity in the world.

The visibility sets together form a connectivity graph specifying the general visibility structure of the 3D world. At this point, it is possible to subject the connectivity graph to an optional optimization step. Many visibility sets are redundant or useless and may be removed from the graph; examples include zero visibility leaf nodes as well as nodes whose visibility is subsumed by its neighbors. Care must be taken not to remove nodes so that the graph no longer is connected, however.

The final step of the visibility calculation phase is to identify the *border voxels* for each visibility set, i.e. the voxels that are adjacent to voxels in another set. We know that that in order to travel from one neighbor to another through a specific node, the tour will have to pass at least one voxel in each border set. Again we can optimize the problem (but this time by an approximation); our implementation identifies a single "entry point" border voxel for each neighbor by minimizing its average distance to the other neighbors in the visibility set together with its counterpart border voxel in the neighboring set.

8.4.3 Tour Generation

The stage is now set for generating the tour through the 3D environment. We use a TSP-like formulation of the problem. It is important to remember that it is not necessary to visit all of the visibility sets in the connectivity graph (as in traditional TSP), just enough to cover the all of the landmarks. Thus, we can model our problem as what is known in the literature as a *Generalized Traveling Salesman Problem* (GTSP), where the n nodes in the undirected graph G are partitioned into m disjoint subsets called *clusters*, and where it is sufficient to visit only one node in each cluster.

Given that GTSP reduces to TSP when $m = n$, GTSP is clearly NP-hard, so in our implementation we do not aim for an optimal solution of the problem. Instead, we use the border voxels computed in the previous phase to reformulate the connectivity graph as a *border graph* with the border voxels as nodes and the interior of the visibility sets as edges. The length of the edges connecting border voxels can either be found using a shortest-path algorithm such as A^* or simply approximated by the Euclidean distance. See Figure 8.5 for an example of a simple border graph for 3D world represented by four visibility sets (A, B, C, and D) and with the paired border voxels as white boxes.

Using this representation, we can now employ a standard TSP heuristic [34] based on computing the minimum spanning tree of the connectivity graph and deriving a Hamiltonian cycle from it. Our unique modification is the added termination condition to quit when all landmarks have been visited. In Figure 8.5, with a starting point in visibility set A, it is easy to see that a tour would proceed in the following order: A, C, B, C, D, C, A.

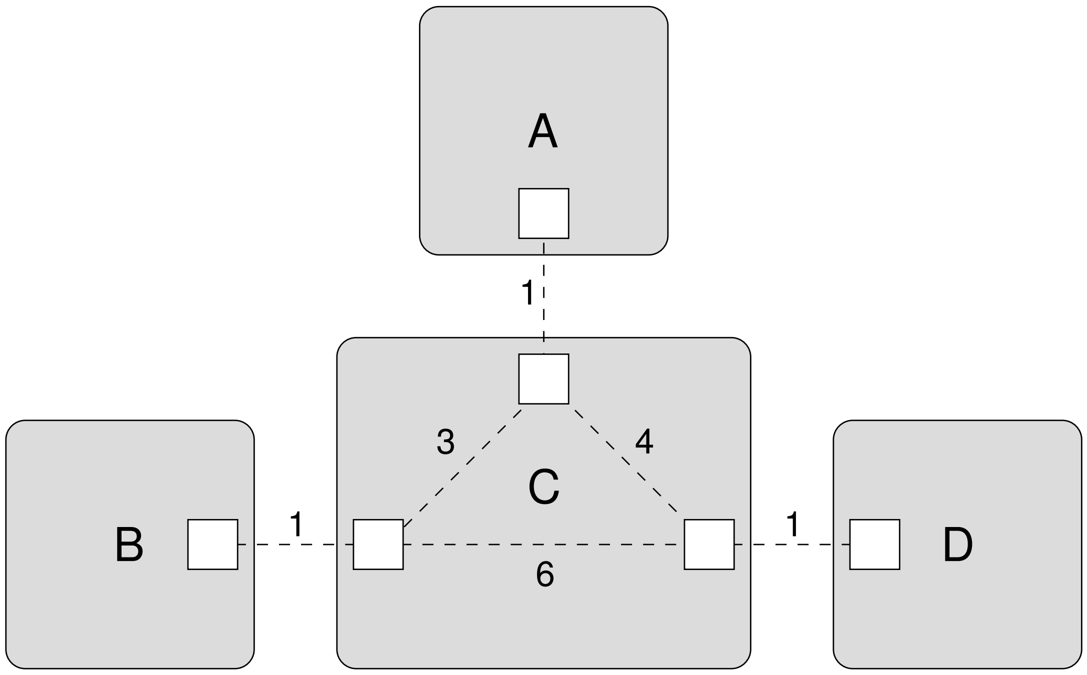


Figure 8.5: Border graph representation of a 3D world.

Finally, the last step of our tour generation phase is to derive a detailed voxel-level tour given the connectivity graph tour computed in the previous step. We do this by iteratively moving along the graph tour, calculating the shortest path from the current position to any border voxel in the next visibility set to visit. Each such instance is constrained to the particular visibility set, cutting down the search space considerably.

8.4.4 Performance

Performance measurements of the tour generation phase applied to the four different scenarios from Section 8.6.5 are presented in Table 8.1. The measurements were conducted on an Intel Xeon 3 GHz computer with 1 GB of RAM. The main bottleneck of the algorithm is the last step, i.e. the derivation of local paths within the voxel sets. Currently, this is performed using a variant of Dijkstra's shortest path algorithm, but more complex and optimized solutions are certainly possible.

As can be seen from the results, the visual complexity of the scene is more or less irrelevant; the voxelization phase is a very small fraction of the total time. Rather, the important metric is the degree of occlusion in the world. For the indoor scenario, the occlusion is high despite high visual complexity, resulting in fast computation. For the outdoor scenario, on the other hand, its open nature yields very large visibility sets, causing high computation time.

The voxel size can be used to somewhat control both computation time as well as memory consumption; the larger the voxels, the shorter computation time and the less memory is used. On the other hand, larger voxel size implies a less accurate volume representation, causing the quality of the generated tours to suffer.

Scenario	**Triangles**	**Time**	
outdoor	558,130	9 minutes	4 seconds
indoor	484,673		59 seconds
infoscape	12,844	7 minutes	49 seconds
conetree	16,576	2 minutes	1 second

Table 8.1: Tour generation performance for the four scenarios.

8.5 Guided 3D Navigation

Our method for 3D navigation guidance is designed to both help the user in discovering all of the specific landmarks in the world, as well as helping her in the building of an accurate cognitive map of the world as a whole. To achieve the former, we employ a grand tour of the world, either created manually by a human designer or generated automatically by an algorithm such as the one described above. To achieve the latter, we allow the user to retain some control over her movement along the tour, seeking to engage the user as an active participant in the exploration.

We call our guided exploration technique *spring-zooming*, inspired by the spring-like umbilical cord that the viewpoint is connected to the grand tour with. See Figure 8.6 for a schematical overview. Depending on the level of interaction desired, we can impose variable constraints on the following properties:

- **Speed.** Movement along the tour can either be computer-controlled or user-controlled—in our implementation, the up and down arrow keys are used to start and stop movement forwards or backwards along the tour.

- **Viewpoint direction.** The direction of the camera can either be slaved to the direction of movement, fixed to follow the currently closest landmark, or fully user-controlled (hybrids are possible).

- **Local deviation.** To facilitate active participation, we can allow deviations from the tour path using the spring-zooming technique. Using a simple interaction technique, the user can smoothly zoom the viewpoint forward or backwards in the direction of viewing to the full extent of the connecting spring (using the center and right mouse buttons in our implementation).

Depending on whether collision detection is enabled or not, the viewpoint may either collide when it comes into conflict with world geometry, or it may float through the geometry as if it was not there. These two events, called collisions and ghosting, may be potentially disorienting to the user, and is typically a major complaint when exploring a 3D world. Avoiding these occurrences is a secondary objective of the spring-zooming technique, and it is done by computing the amount of free space around the tour in all points and constraining the full length of the umbilical cord to this value. This ensures a smooth and continuous ride through the environment with no jarring stops or confusing ghosting.

The grand tours accepted as input are generally discrete waypoints in space, and so we fit Hermite curves [56] to these points to smooth the movement through the 3D space.

8.6 User Study

The basic premise of this research is that guiding the user in exploring a 3D world will increase the user's efficiency in solving visual search tasks compared to unguided navigation. However, we also hypothesize that fully constraining the movement of the viewpoint will reduce the viewer to a passive recipient instead

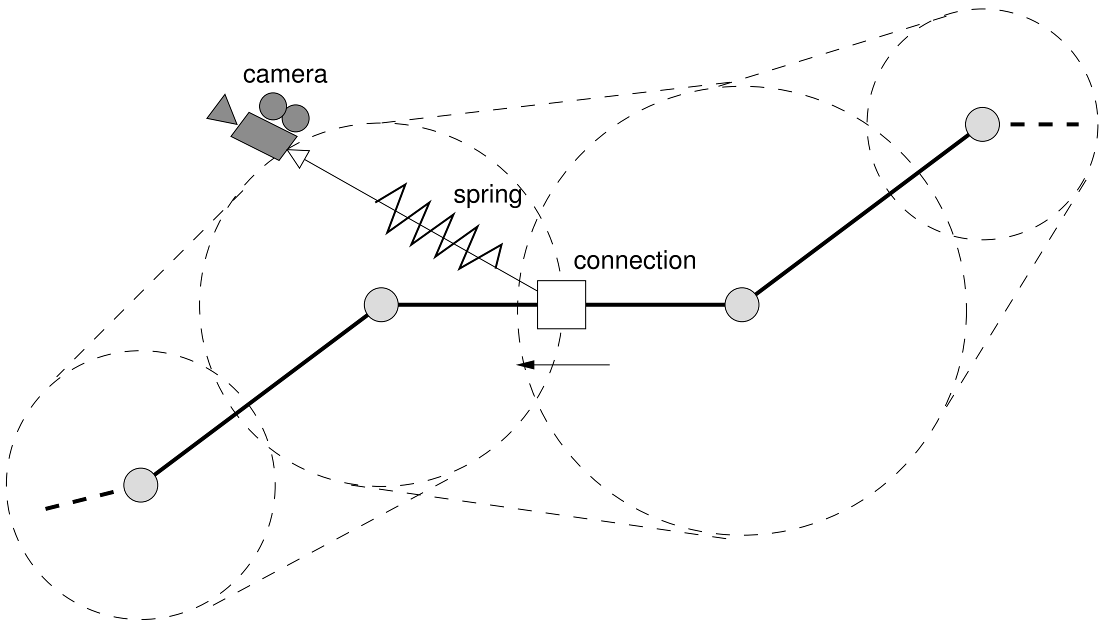


Figure 8.6: Spring-zooming overview (the circles show the free space around each node.)

of an active participant, somewhat akin to being a passenger in a car as opposed to driving the car yourself. Accordingly, the user's perception of the world as a whole will suffer even if he or she is shown the important landmarks by the guidance technique.

To evaluate these two statements, we conducted a formal user study exposing a number of subjects to visual search tasks in four different types of environments with a recall phase designed to test general familiarity with the environment and an evaluation phase for measuring visual search performance.

8.6.1 Subjects

We recruited 16 subjects for this study, four of which were female. The subjects were all undergraduate and graduate students from the engineering programs at our university. Ages ranged from 20 to 50 years. All participants were screened to have at least basic computer skills, were not color blind, and had normal or corrected-to-normal vision. 9 out of 16 subjects had extensive 3D experience.

8.6.2 Equipment

The experiment was conducted on an Intel Centrino Duo laptop computer equipped with 2048 MB of memory running the Microsoft Windows XP operating system.

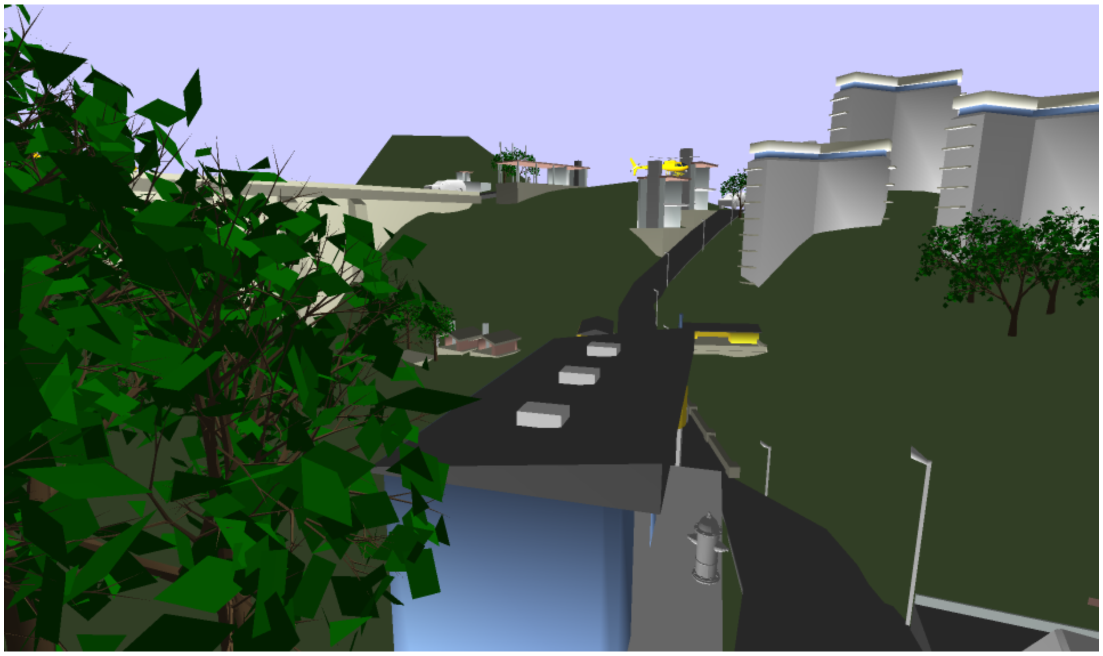


Figure 8.7: Outdoor scenario.

The display was a 17-inch widescreen LCD display running at 1920 × 1200 resolution and powered by an NVidia Geforce 7800 GO graphics card.

8.6.3 Procedure

Each test session lasted approximately one hour, and started with an introduction and a training session to allow the subject to get familiarized with the test application, the interface, and the test procedure. After the subjects indicated that they were satisfied, we proceeded with the actual scenarios. In order to increase the generality of the study, we included four different scenarios designed to mimic various contexts where navigation guidance may be used.

Each scenario was tested in a three-phase sequence: familiarization, recall, and evaluation. In the first phase, users were given the scenario world and were allowed to familiarize themselves with it for five minutes. During this phase, the actual guidance method selected for the user was active. The subject was given a reference card with pictures of three types of landmarks relevant to the actual scenario that he or she should be looking for. An overhead map of the world with the user's own position and location marked was available in the upper left corner of the display.

After five minutes, the experimenter moved on to the recall phase, where the subject was shown a full-screen overhead map of the world and was asked to place as many instances of two of the three target landmarks they could remember.

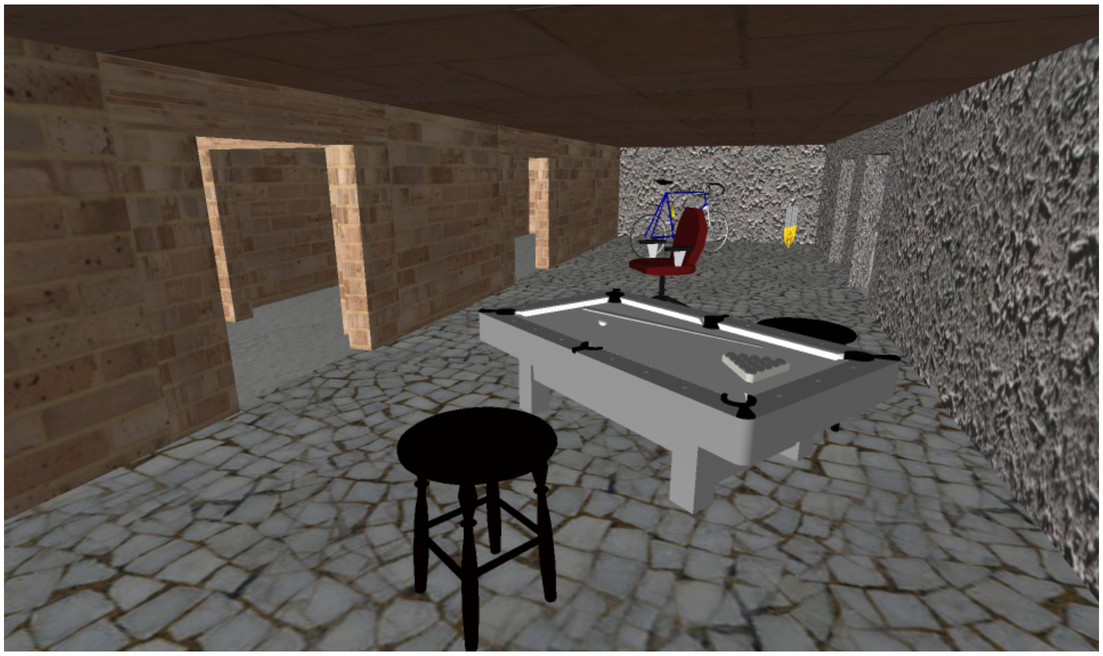


Figure 8.8: Indoor scenario.

There was no time limit here.

Finally, in the third phase, the subject returned to the 3D world with the task to collect as many as possible of the third type of landmark. Here all subjects were forced to navigate freely with no guidance support. Collecting an object was done by approaching to within a distance of 5% of the world scale and pressing the Tab key. This removed the object from the landscape. The miniature overhead map was available in this phase as well. When the subject decided that all targets had been found, he or she was able to end the scenario (stopping the time).

Subjects received the four scenarios in counterbalanced order to manage systematic effects of practice. The subjects did not know in advance which two of the target types they would be asked to place in the second phase, nor which landmark to collect in the third.

8.6.4 Navigation Methods

The navigation method employed was one of the following three:

Free. Unaided first-person 3D navigation with no guidance. The mouse panned the view and the arrow keys moved in the direction of viewing (left and right for strafing).

Follow. Passive tour following with full guidance except for camera orientation. The mouse panned the view.

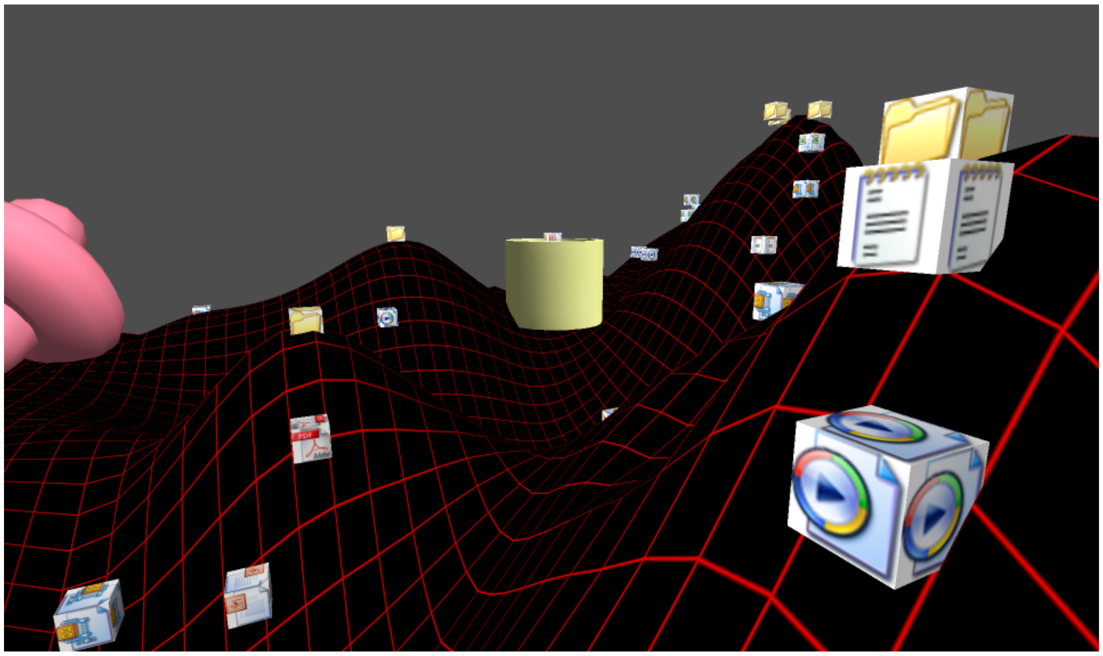


Figure 8.9: Infoscape scenario.

Spring. Full spring-zooming with user-controlled movement, deviation, and camera orientation. The mouse panned the view, the center and right mouse buttons engaged forward and backward zooming, and the up and down arrow keys controlled movement along the tour.

The free navigation method was used for all subjects in the third phase (evaluation).

8.6.5 Scenarios

The four different scenarios employed in the experiment were designed to depict typical usage situations of 3D worlds and 3D navigation using both abstract as well as realistic environments. Subjects were given a concrete explanation of the scenario prior to starting each scenario run. Below follows a short description of each scenario (see the example screenshots in Figure 8.7 through 8.10):

Outdoor. Large-scale outdoor world with a rescue mission scenario where the user was asked to identify helicopters, cars, and fire hydrants. [realistic]

Indoor. Maze-like single-floor indoor environment representing a furniture store where the user was looking for office chairs, sofas, and floor lamps. [realistic]

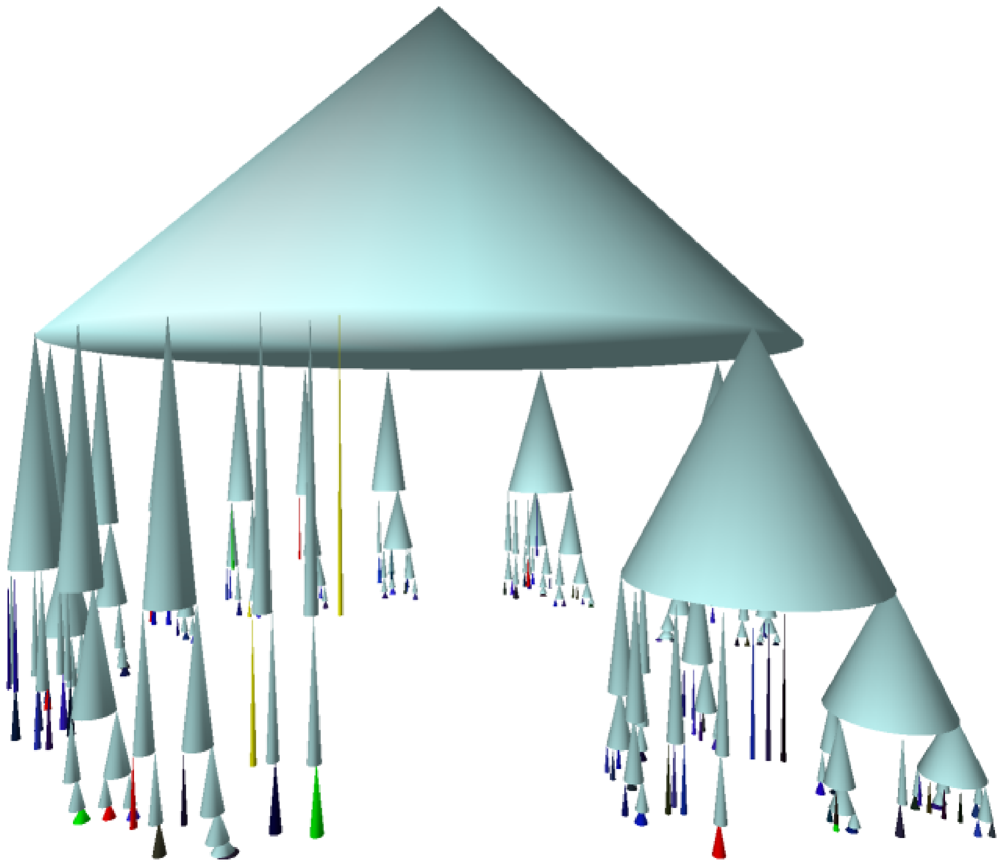


Figure 8.10: Conetree scenario.

Infoscape. Abstract information landscape for a hypothetical 3D file browser where the subject was preparing for writing a report by looking for Word, PDF, and Excel files. [abstract]

Conetree. Abstract conetree [120] visualization for the organization hierarchy of a company where the subject was asked to look for leaf nodes of specific colors. [abstract]

8.6.6 Design

The experiment was designed as a between-subjects comparative study on the independent variable Method (the navigation method used), and with a within-subjects independent variable Scenario (the type of environment, see the previous section). The method variable used the three levels from Section 8.6.4, i.e. “free”, “follow”, and “spring”. The dependent variables were the relative error (i.e. the number of missed landmarks divided by the total number of landmarks) and average error distance for the recall phase, and the relative error and average time per found landmark for the evaluation phase.

After having completed a full set of tasks, we instructed the subjects to fill out a post-test questionnaire asking them about their experiences during the test.

8.7 Results

The results from the user study can be divided into performance results, and subjective ratings.

8.7.1 Performance

In general, the results from the user study confirmed our belief that subjects with navigation guidance would be more efficient at solving the visual search task than those without guidance: the time per found landmark was 43.8 (s.d. 30.4) seconds for free navigation, 19.5 (s.d. 10.7) seconds for follow navigation, and 17.6 (s.d. 11.9) seconds for spring-zooming (see Figure 8.11). This difference was also significant: $t(42) = 3.398, p = .001$ for the comparison between free and follow, and $t(42) = 3.624, p = .001$ for free versus spring. The difference between follow and spring was not significant ($t(38) = .529, p = .600$).

Furthermore, subjects using spring-zooming were also more correct in collecting landmarks in the evaluation phase than those using free navigation: the error rate was .198 (s.d. .245) for free navigation, .123 (s.d. .160) for follow mode, and .058 (s.d. .130) for spring-zooming. The difference was only significant for free versus spring ($t(42) = 2.297, p = .027$) and not for free versus follow ($t(42) = 1.179, p = .245$) nor for follow versus spring ($t(38) = 1.400, p = .170$).

What was more surprising was that subjects were more accurate in placing landmarks in the recall phase for spring-zooming than for the other two methods: the average error distance per landmark (normalized using the scale of the world) was .236 (s.d. .124) for free navigation, .192 (s.d. .102) for passive follow navigation, and .099 (s.d. .073) for spring-zooming. The difference was significant between free and spring ($t(42) = 4.335, p < .001$) as well as follow and spring ($t(42) = 2.297, p = .027$), but not between free and follow ($t(38) = 3.282, p = .002$). This goes against our hypothesis that the navigation methods which permit some measure of user control (i.e. free navigation and spring-zooming) would promote significantly better recall than passive tour following; as it turned out, spring-zooming was significantly more accurate than both other methods.

Finally, as for correctness in the recall phase, no conclusive results were found: the relative error rate was .238 (s.d. .191) for free navigation, .174 (s.d. .173) for follow mode, and .191 (s.d. .201) for spring-zooming (see Figure 8.12). None of these differences was significant: $t(42) = 1.146, p < .258$ for free versus follow,

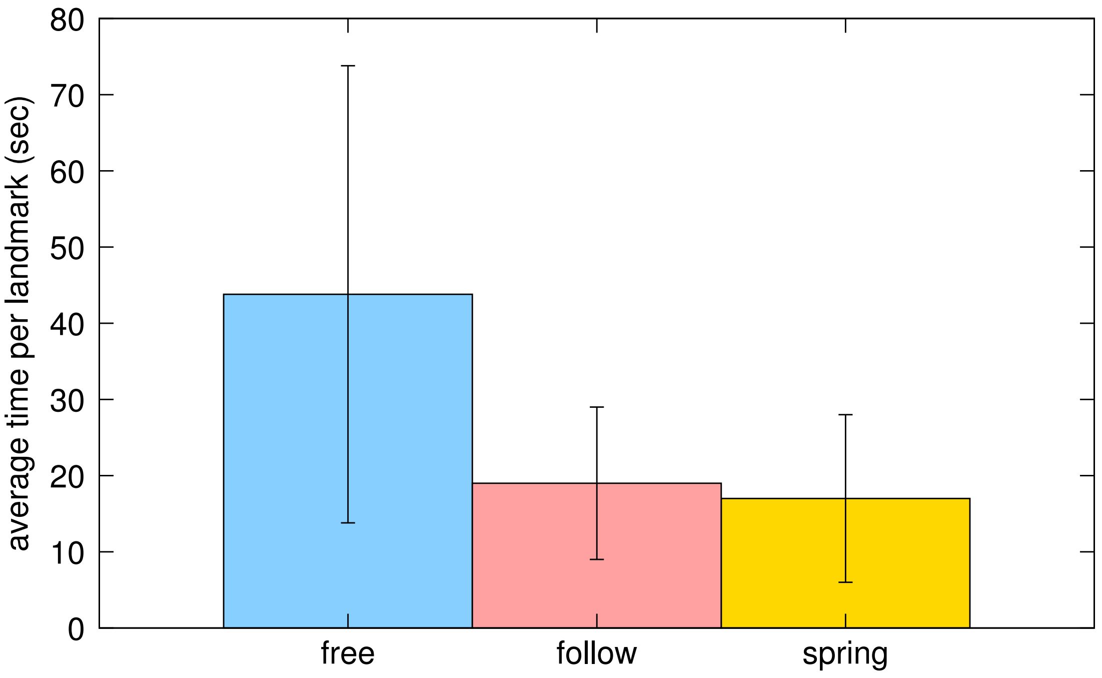


Figure 8.11: Performance results for the evaluation phase (average time per found landmark). Error bars show standard deviations.

$t(42) = .782, p = .439$ for free versus spring-zooming, and $t(38) = -.289, p = .774$ for follow versus spring-zooming.

8.7.2 Subjective Ratings

The results from the subjective ratings were inconclusive: the subjects disagreed on the difficulty of most parameters, including ease-of-use, enjoyability, and efficiency. Furthermore, there were few conclusive results from their self-reported performance in the recall and evaluation phases.

We did find significant differences for the follow versus spring-zooming methods using Kruskal-Wallis tests down to $p = 0.5$ for the ease-of-use (2.40 (s.d. .548) for follow, 3.60 (s.d. .894) for spring-zooming), sense of orientation for the conetree (2.80 (s.d. .837) for follow versus 1.80 (s.d. .447) for spring-zooming), and for locating objects in the recall phase for the indoor environment (2.20 (s.d. .447) versus 3.80 (s.d. .837)).

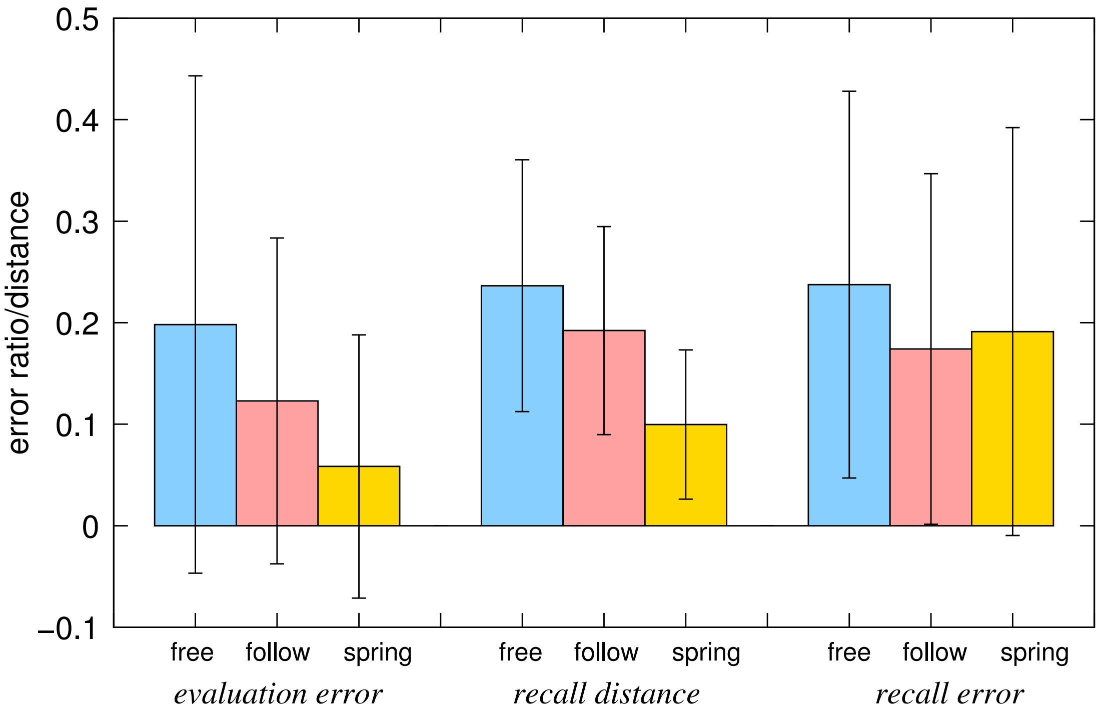


Figure 8.12: Correctness results (evaluation error rate, recall error distance, recall error rate). Error bars show standard deviations.

8.8 Discussion

While the results from the user study confirmed our basic hypothesis that navigation guidance will improve search performance over free navigation, it was a little bit surprising that our second hypothesis on user control promoting the formation of a cognitive map was not confirmed. One possible explanation might be that the subjects in the passive follow group were not in fact passive recipients since they were given a very specific task when familiarizing themselves with the 3D world. Therefore, they performed better than they might have done without this knowledge. However, our pilot testing showed that the alternative, i.e. not telling the subjects which kinds of landmarks to look for, was simply not feasible for the high-detail scenarios we used in the study.

Regrettably, two of the participants in the user study became motion sick (one still finished the study, the other was forced to cancel). An interesting observation is that both of these participants were assigned to the passive tour following group—a plausible (if perhaps unfounded) explanation may be that users that have no control over their movement run a greater risk of this, somewhat akin to how people who are prone to motion sickness while riding cars typically only get it when they are passengers and not driving themselves.

Only a summary of the results were presented in this paper. We have also analyzed the results based on the scenario, and these indicate the same general trends as the overall results. It is worth noting that there were no significant differences between any of the dependent variables in the indoor scenario, an environment with a high degree of occlusion. For the conetree scenario, on the other hand, an environment with a low degree of occlusion, the average time per target was not significant for any method (recall distance still was for spring-zooming, however).

8.9 Conclusions

We have presented a method for navigation guidance in the exploration of general 3D environments intended to both promote the user's building of a cognitive map of the environment as well as to improve visual search task performance. The method works for both abstract as well as realistic visualizations and operates in two distinct steps: an off-line tour generation step that builds a grand tour of the given 3D world that visits all of its landmarks, and an on-line interactive navigation technique that guides the user along the tour while still allowing for some user control. This last step is vital in order to make the user an active participant in the navigation. A user study was conducted to investigate the impact of the new technique compared to free navigation as well as passive tour following, and the results indicate a significant improvement for both search performance and general recall for the new technique.

Acknowledgments

Thanks to Robert Karlsson for supplying some of the 3D models that were used in the user study scenarios. Thanks to the anonymous test subjects who participated in the study.

Chapter 9

Conclusions and Future Work

Making sense of a computer-generated 3D world is hard. If anything, the empirical studies conducted as part of this thesis have shown this fact to be true time and time again. Furthermore, it does not only hold for desktop computers, but also for some of the best immersive Virtual Reality systems money can buy. Even when exploring our 3D environments in what must be considered the state-of-the-art when it comes to immersion and presence, subjects performed relatively poorly, were easily disoriented, and some even experienced motion sickness and nausea. It is clear that much remains to be done in perfecting the AR/VR/MR devices of the future.

The techniques presented in this thesis go a little way towards making visual tasks simpler to perform and more powerful to use in a three-dimensional environment. In our work, we have striven to build a comprehensive toolbox of different techniques ready for adoption into both existing as well as newly developed 3D applications. Some of them, like the view projection animation technique, are non-invasive in nature and require very few changes to existing code; others, such as our approach to navigation guidance, more or less dictate the structure and interaction model for a whole application.

Nevertheless, much effort has been invested to make the different techniques as orthogonal as possible, utilizing more or less disjoint subsets of the solution space to manage occlusion. The ambition has been that a few, all, or just two of the techniques should be possible to combine at the same time. In this venture we have succeeded; there are no conflicting factors in any of the four occlusion management techniques that would prevent them from being used together. In fact, it would be an interesting exercise to actually put all four techniques together into a single visualization platform and see the possible synergies between them. This is left for the future, however.

Beyond the practical work of designing, implementing and evaluating the four techniques described here, the work presented in this thesis was designed from the beginning with a firm theoretical model in mind. The goal to develop a toolbox of "occlusion reduction" techniques, as we originally called it, was clear from the start. The creation of a taxonomy capturing this particular subset of 3D interaction techniques helped mold this vision and make us understand it better. Identifying five different design patterns from the body of existing work in the field can almost be seen as the crowning moment of this research; albeit possibly something of a self-fulfilling prophecy, it was nevertheless encouraging to see the hierarchical clustering process confirm the five patterns we had informally been thinking about ever since the beginning of the project.

Four different techniques means four different empirical user studies measuring the performance of our new methods compared to, mainly, standard unaided 3D navigation. Our ambition, with the exception of the BalloonProbe study conducted in Chapter 6, was never to compare our techniques to existing occlusion management techniques. This was mainly because little relevant work existed in the area, and most of the relevant work was not developed with our particular goals for occlusion management in mind. This is a definite area of improvement for the future, however. Nevertheless, we believe that the results from our user studies quite comfortably show that our new methods, every one of them, does improve the efficiency of performing visual tasks in 3D environments, and we expect them to perform well in comparison to other occlusion management techniques.

While the topic of occlusion management is far from exhausted, for the future it might be beneficial to turn our eyes towards an even more general formulation of the problem. Why contend ourselves with merely reducing occlusion when there is a host of other visual cues to manipulate? We call this approach *augmented perception*, i.e. the management of visual cues in order to amplify the perceptual abilities of the human user. We believe that there is much to be won by weakening (or strengthening) certain visual cues to help the user pinpoint important information. This method would mainly be orthogonal to visualization techniques, allowing us to combine our methods with both new and existing work. Perhaps we can extend the basic idea of our dynamic transparency work (see Chapter 7) to create "superhero vision" on an even grander scale? The consequences of such work would certainly be intriguing.

Part II

Causality Visualization

Chapter 10

Introduction

In modern use, the notion of causality is associated with the idea of something (the cause) producing or bringing about something else (its effect). The term "cause" has a broader meaning and is used as an explanatory or reasoning tool. Identifying causal relations in a complex system can be the first step towards understanding the underlying mechanisms that determine the system's laws. Therefore, causal relations cover a wide variety of scientific domains.

Our interest in causality originates mainly from the viewpoint of distributed and parallel computing, where causal relations are used extensively for example (i) in distributed database management to determine consistent recovery points; (ii) in distributed software systems for determining deadlocks; (iii) in distributed and parallel debugging for detecting global predicates and detecting synchronization errors; (iv) in monitoring and animation of distributed and parallel programs to determine the sequence in which events must be processed so that cause and effect appear in the correct order; and (v) in parallel and distributed software performance to determine the critical path abstraction.[1] Improving the graphical visualization of causal relations will thus benefit all these activities.

Causality is a much broader concept than this, however, and is not restricted to computer science research; examples include social networks, biology, information theory, etc. In an effort to show this (and to make use of the new visualization techniques in a real-world application), we have conducted a case study of the use of our techniques for the visualization of large-scale citation networks. See Chapter 13 for more information on this.

In this chapter, we give a brief background to the causality visualization problem, including a brief formal introduction to causal relations, a description of the various analysis tasks involved when studying causal relations, and a presentation of the existing work in the area.

[1]The longest sequential thread, or chain of dependencies, in the execution of a parallel or distributed program.

10.1 Definitions

A causal relation is the relation that connects or relates two items, called *events*, one of which is a cause of the other. Obviously, for an event to cause another, it is not sufficient that the second merely happens after the first; however, it is well accepted to state that this is necessary, and temporal order can be relied on to explain the asymmetrical direction of causal relations.[2] All events connected in the causal relation are part of a set of *processes*, labeled $P_1, \ldots, P_N$, each of which can be thought of as a producer of a disjoint subset of the set of all events in a system. Events performed by the same process are assumed to be sequential; if not, we can split the process into sub-processes. Thus, it is convenient to index the events of a process P_i in the order in which they occur: $E_i = e^i_1, e^i_2, e^i_3, \ldots$

For our purposes, it suffices to distinguish between two types of events: *external* and *internal* events. Internal events affect only the local process state. An internal event on process P_i will causally relate to the next event on the same process. External events, on the other hand, interconnect events on different processes. Each external event can be treated as a tuple of two events: a *send* event and a corresponding *receive* event. A send event reflects the fact that an event, that will influence some other event in the future, took place and its influence is "in transit". A receive event denotes the receipt of an influence-message together with the local state change according to the contents of that message. A send event and a receive event are said to correspond if the same message m that was sent in the send event is received in the receive event.

We now formally define the binary causal relation $\rightarrow$ over all the events of the system E ($\rightarrow \subseteq E \times E$) as the smallest transitive closure that satisfies the following properties [89]:

1) If $e^i_k,\ e^i_l\ \in E_i$ and $k < l$, then $e^i_k \rightarrow e^i_l$.

2) If $e^i = \mathit{send}(m)$ and $e^j = \mathit{receive}(m)$, then $e^i \rightarrow e^j$ where m is a message.

When $e \rightarrow e'$, we say e causally precedes e' or e caused e'. Causal relations are irreflexive, asymmetric, and transitive.

10.2 Analysis Tasks

At the onset of our investigation into visualization of causal relations, we organized a formative evaluation of these concepts using a focus group consisting of researchers working on distributed systems. The evaluation took the shape of a

[2]It has been argued that not even this is necessary, and that both simultaneous causation and "backwards causation" (effects preceding their causes) are at least conceptually possible. This, on the other hand, causes problems when considering the asymmetric nature of causal relations.

panel discussion on questions related to causal relations and their use, and six members of the Distributed Computing & Systems group at the Department of Computing Science at Chalmers participated in the session. These discussions allowed us to identify the typical analysis tasks a user is interested in when studying a distributed system, and were vital in tailoring our visualization to these tasks. Below follows a short overview of these analysis tasks.

10.2.1 Lifecycle Analysis

The lifecycle of an individual process is often of great interest when analyzing a system of causal relations. This includes aspects such as the duration of a process as well as its starting and stopping times (both in isolation as well as in relation to other processes), aspects that are vital in understanding how a system works.

10.2.2 Influence Analysis

The analysis of influences and dependencies in a distributed system was found to be one of the most important analysis tasks when studying the flow of information in a system. Designing, debugging, or trying to grasp the underlying mechanisms of a distributed system or algorithm all involve this task.

10.2.3 Inter-Process Causal Relations

Often, a practitioner studying a system of causal relations needs to know whether two nodes, P_i and P_j, in the system are causally related, i.e. if there exists an event $e^i \in E_i$ and an event $e^j \in E_j$ such that $e^i \rightarrow e^j$. Of course, this causal relation can go through several levels of transitive indirection, and is therefore quite difficult to spot manually or by using Hasse diagrams (as we will see).

10.3 Related Work

There has been surprisingly little work performed in the area of causality visualization, and the prevalent visualization method is still the traditional Hasse (also known as time-space) diagram. Figure 10.1 shows an example of a time-space diagram for a system comprised of three processes, where the progress of each process is described by a directed horizontal line, the process line. Time is assumed to move from left to right. Events are symbolized by dots on the process lines, according to their relative order of occurrence. Messages are shown as arrows that connect send events with their corresponding receive events. Visualizations of causal relations in the form of such time-space diagrams are currently quite standard in visualization and debugging platforms for parallel and distributed systems, and the number of such platforms is too large to allow discussing them

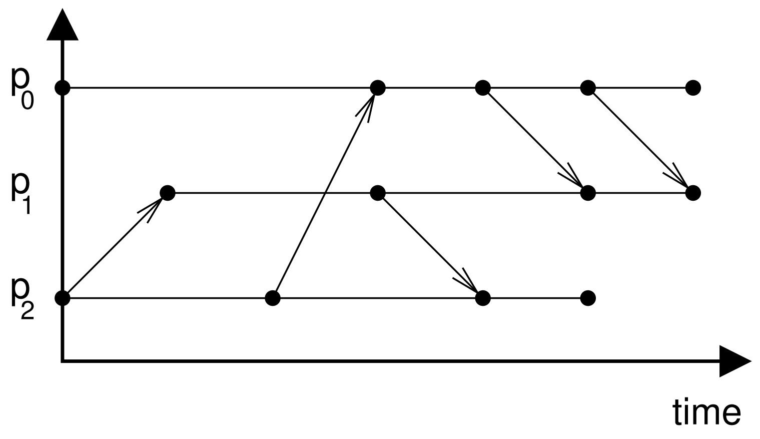


Figure 10.1: Hasse diagram visualization with 3 processes.

all; we will just focus on a few of the noteworthy systems. One of the first of the new generation of visualization tools to include the time-space diagram was the Voyeur [128] system, which provided a framework for defining various animation views for parallel algorithms. The TOPSYS [11] environment includes various standard concurrency visualizations (called VISTOP) integrated with the debugging and performance analysis tools of the system, with time-space visualization being one of them. Using this process-based concurrency view, users can identify synchronization and communication bugs. Going one step further, the conceptual visualization model of the VADE [105] system is based on the causal relation notion. VADE is also geared towards more general algorithm visualization, and supports not only communication events but also other algorithmic objects and events. Also of interest is LYDIAN [86], an educational visualization system, which by default constructs the time-space diagram for every algorithm implemented in the system. Kraemer and Stasko [87] describe the essential characteristics of toolkits for visualization of concurrent executions, and introduce their own system, called Parade. Parade also includes an animation component called the Animation Choreographer that orders display events from a trace file in much the same way as the techniques described in this paper. In addition, for the purpose of our study, the Hasse visualization used in Figure 10.1 is very similar to the time-space visualization view from the ParaGraph system [73, 74] and its adaptation in the PVaniM tool [136], as well as the Feynman or Lamport views from the Polka animation library [131].

While Hasse diagrams certainly are in widespread use, they have a number of deficiencies that decrease their usefulness for realistic systems. First of all, a Hasse diagram offers only local dependency information for each process and not the transitive closure of all interactions involving it, making it difficult to gain an overview of the overall information flow in the system; in essence, the user is forced to manually backtrace every single message and process affecting a specific process to find its dependencies. Second, the fine granularity of the visualization makes Hasse diagrams difficult to use for large systems of ten or more involved nodes; the amount of intersecting message arrows simply becomes too overwhelming for complex executions. Third, Hasse diagrams are intrinsically static in nature and thus make little use of the interactivity of the computer medium; animation and creative use of color are likely to be useful tools in this kind of visualization.

Ware et al. [142] presented a new visualization construct called a *visual causality vector* (VCV) that represents the perceptual impression of a causal relation and employed animation to emphasize this relation in a directed acyclic graph. Three different VCVs were introduced based on different metaphors: the pin-ball metaphor, where the VCV is a ball that moves from the source to the destination node, striking the destination and making it oscillate; the prod metaphor, where the VCV is a rod that extends from the source to prod the destination; and finally a wave metaphor, where the VCV accordingly is an animated wave that moves towards the destination node. However, while these constructs are certainly improvements over simple DAG representations of causal relations, they do nothing to battle the complexity of large systems with many nodes and relations. In fact, Ware's primary contribution is the investigation of timing concerns for the perception of causality for users, not the visualization technique *per se*. It might still be interesting to incorporate Ware's VCVs into our system in some form.

10.4 The CausalViz Framework

In order to test the Growing Squares and Growing Polygons techniques and to be able to perform user studies on their effectiveness, we implemented a general application framework for the visualization of causal relations called CausalViz (see Figure 10.2). The framework is implemented in C++ on the Linux platform and uses the Gtk+/Gtk– widget toolkits for user interface components as well as OpenGL for graphical rendering.

10.4.1 System Architecture

The architecture of the CausalViz application (see Figure 10.3) is based around a single partially ordered set (*poset*) representing the execution data under study. A number of visualization components observe this set and present graphical representations of the data (potentially allowing the set to change during runtime). There currently exist three different visualizations, i.e. traditional Hasse diagrams, the Growing Squares visualization, and the Growing Polygons visualization.

Central in the system architecture is the *application manager* that creates all

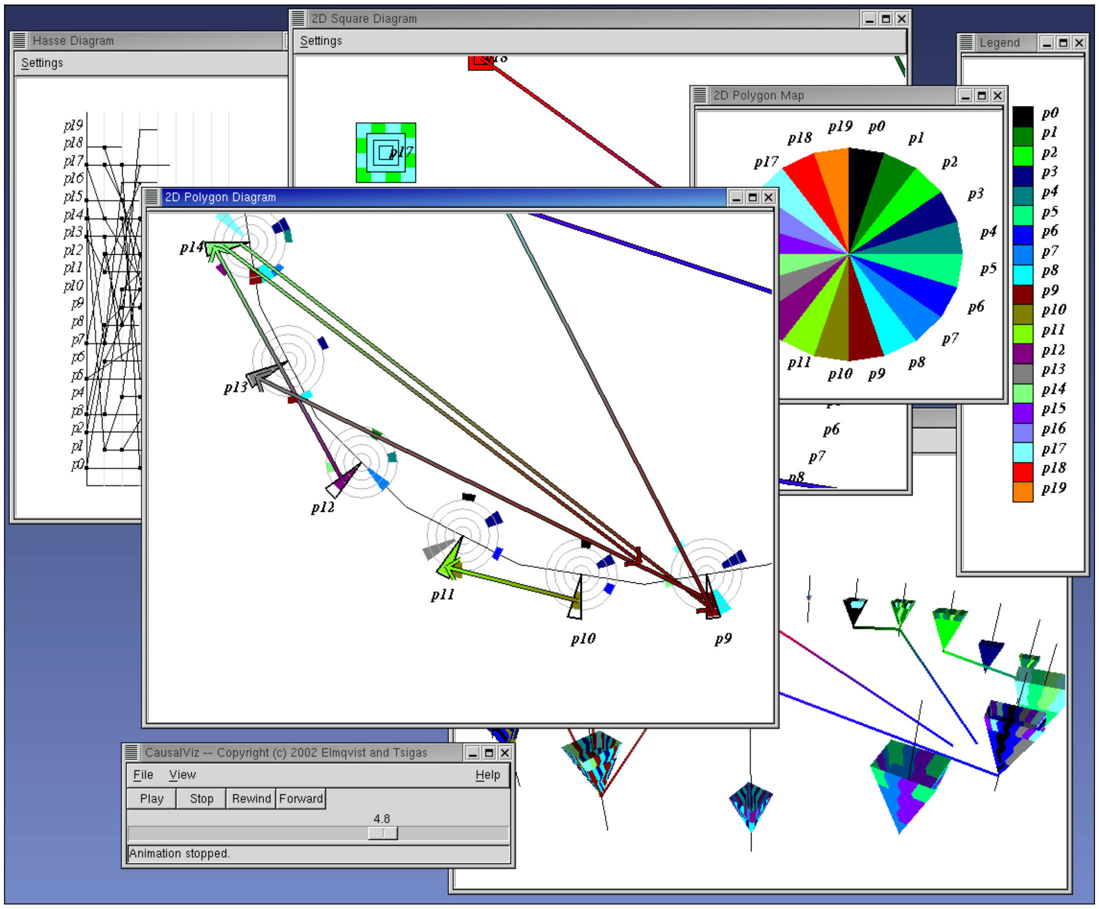


Figure 10.2: The CausalViz application.

the other components, manages the graphical user interface (GUI), and performs loading of data files into the application (stored in a general XML format for partially ordered sets). In order to allow for the animation of events in the visualizations, there also exists a general *animation manager* thread that the visualization components can use to smoothly interpolate values in the poset with respect to time.

10.4.2 Poset Management

System execution traces are stored in a general XML file format for partially ordered sets. Here, a process P_i is represented by the subset $E_i \subseteq E$ of all the events in the system belonging to the process and a set of messages M_i. Messages are partial orderings between events in different subsets (processes), and can thus be represented by pairs of events, i.e. $M_i \subseteq E \times E$. It is then up to the application to compute the minimal transitive closure for the poset.

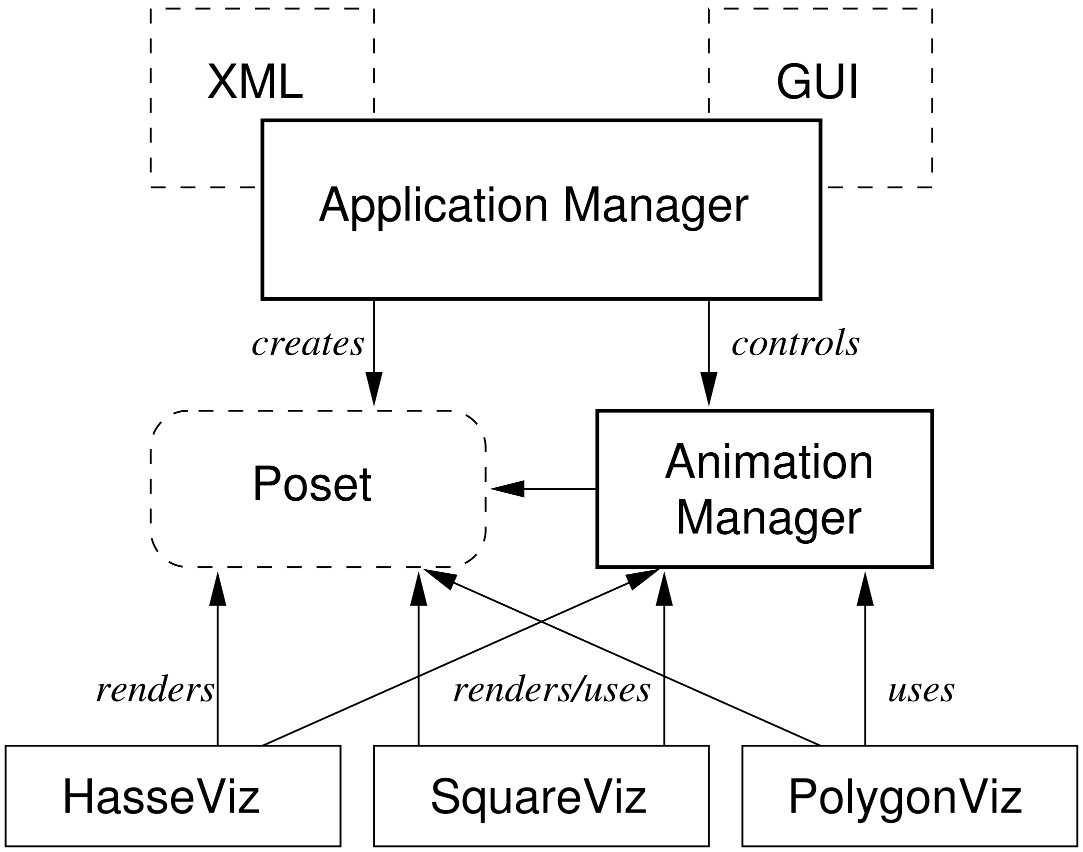


Figure 10.3: CausalViz system architecture.

In the CausalViz application, the transitive closure is computed using a modified topological sort [34]. The objective of the algorithm is two-fold: (i) to derive the transitivity information for each event (i.e. the processes that have influenced it so far) and (ii) to assign the event to a discrete time slot. This is done by greedily consuming sequential events in each subset (i.e. process) of the poset until reaching an event with unresolved dependencies (i.e. a partial ordering to a previously unvisited event). When this happens, the algorithm moves on to the next process to continue from where it last left off. This is repeated until all events in the system have been visited. The current influence of each event is easily maintained and updated during this process, and illegal cyclic dependencies are trivially detected by checking whether the algorithm has cycled through all process without visiting any new events.

10.4.3 CiteWiz Extensions

The CiteWiz application described in Chapter 13 is largely based on the CausalViz framework with a few extensions. Instead of using a partially ordered set as the main data structure, CiteWiz uses a citation database and a user-defined hierarchical view of this database that is then used for visualization.

The implementation of the Growing Polygons visualization used in CiteWiz is the same as the CausalViz implementation, with some modifications to improve the scalability of the technique for large and highly connected networks (see Chapter 13 for details on these modifications).

Chapter 11

Growing Squares

As described earlier, there is surprisingly little work on visualization of causal relations besides various implementations of Hasse diagrams, a fact that is especially curious in light of its shortcomings for understanding a distributed system. The fine granularity of Hasse diagrams defeats their use as overview tools, and they transfer the burden of maintaining transitive relations to the user herself. This means that a user studying the information flow in a distributed systems visualized using a Hasse diagram might potentially have to backtrace every single message and process in order to get a clear picture of the influences in the system.

The Growing Squares visualization technique (first presented in [46]) was designed to help the user quickly get an overview of the causal relations in a system by making use of animation, color and patterns in an intuitive way. The visual metaphor of the technique is that of "pools" of color spreading on a piece of paper as time progresses, each color and pool representing a specific process or node in the system. Messages in the system are shown as "channels" from one pool to another. Each color pool will start growing at the time its corresponding process is started, and accordingly stop growing when the process stops executing events. The channels representing messages from one process to another intuitively carry the color of its source with it, resulting in the destination pool receiving this color as well. However, like age rings on a tree, the color of the new influencing process will only be present in the destination process starting from when the message was received.

Figure 11.1 gives an example of a system with two processes, P_0 and P_1, colored blue and white, respectively. The color pools are represented as 2D squares that grow over time. At a certain time t, P_0 sends a message to P_1 (denoted by the arrow in the figure), establishing a causal relation between P_1 and P_0. For all times $t' > t$, the color pool of process P_1 now shows this influence from the blue P_0 by means of a checkered pattern combining the two colors.

In order to visualize the transitive property of the causal relation (see the previous section), a similar color pattern scheme is used. In Figure 11.2, process P_1 is sending a message to P_2 (colored red) *after* having been influenced by a

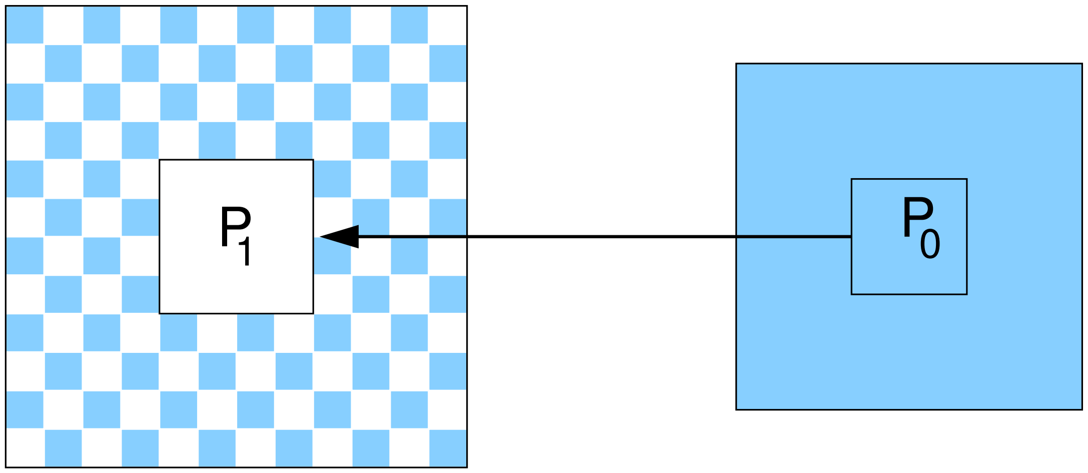


Figure 11.1: Simple example of the Growing Squares technique with two processes.

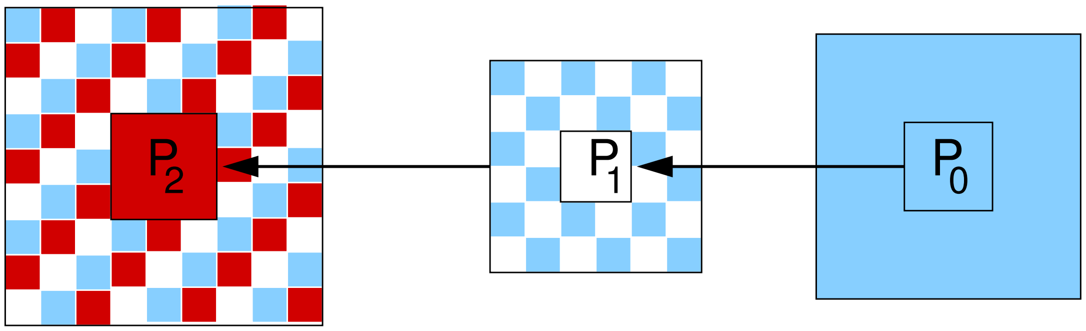


Figure 11.2: Transitivity property of causal relations using Growing Squares.

message from P_0. Now, both the color of the source process (white from P_1 itself) and any of its existing influences at the time of sending the message (blue from P_0) are transferred to P_2, making its texture from this time and onwards a checkered pattern of all of the three colors. It is now easy to see that P_2 is causally related to both P_0 and P_1.

Multiple influences from the same source process will increase the amount of the source process's color in the texture of the destination process. Even if the checkered pattern makes it difficult to see the exact ratio, this fact can nevertheless be used as a visual indication that multiple influences have occurred.

Having abandoned a traditional timeline, the Growing Squares method is dependent on animation to allow the user to view the entire execution of the system under study. Starting at $t = 0$, the user can advance the time in the system to observe the system execution in chronological order, or choose to view the situation at specific points in time. This is another radical difference from Hasse diagrams; Hasse diagrams are static in nature and do not benefit much from animation, whereas Growing Squares are dynamic and rely on animation to present the full data set to the user.

Figure 11.6 shows an example sequence consisting of five processes in a distributed system visualized using the Growing Squares technique. The state of the visualization is here shown for each discrete time unit (in practice, the animation is fluid and continuous between the time steps) starting at $t = 1$ and ending at $t = 5$, the end of the execution. Processes are laid out in a clockwise fashion with P_0 at the top. Screenshot (a) at $t = 1$ shows how P_1 sends a message to P_0, starting it (it has zero size up until this time), and (b) at $t = 2$ depicts the two colors (green and black) in the process square of P_0. In the same screenshot, P_4 sends a message to P_1, causing P_1 in (c) at $t = 3$ to hold influences from both P_4 as well as indirectly from P_3 (i.e. an example of the visualization of transitivity in the Growing Squares visualization). In (d) at $t = 4$, two messages originating from the otherwise isolated P_2 reach P_0 and P_4, its blue color showing in the outer square of these two processes in snapshot (e) at $t = 5$.

11.1 Design

In order for the Growing Squares visualization to be effective, users must be able to distinguish between the individual process colors easily in the system under study. Selecting a suitable color scale is thus an important aspect of the method, and we investigated the use of perceptually uniform color scales such as LOCS [90, 91] for this purpose. However, we found that the continuous nature of LOCS was not well suited to our problem since it made distinguishing between adjacent colors difficult, and the scale itself included an inordinate amount of dark colors. Instead, we opted for a simple color scale with the individual colors uniformly distributed over the RGB spectrum.[1]

One of the central features of the presented visualization technique is that it draws process squares with the checkered patterns containing all the colors of the processes that have influenced the process. If the number of influences is large, the on-screen space allocated for each color will be very small and thus hard to distinguish (see [146] for in-depth information on color perception). In order to allow the visualization to continue to be effective, we need a zoom function that allows the user to view the graphical representation at different magnification levels effortlessly. We have implemented a simple continuous zoom mechanism for this purpose; in the future, it may be extended to borrow techniques from the Pad [110] zoomable user interface and its descendants [9, 10].

[1]The RGB color model was chosen for simplicity; a color model like HSV might be more suitable to human perception.

It might be argued that using circles instead of squares would have been more in keeping with the metaphor of color pools spreading on a piece of paper. Our original intention was also to use circles, but we ultimately chose squares for a number of reasons: (i) the larger area of squares facilitates color recognition better than circles, (ii) the layout of process squares into grids is easier (no wasted space), and (iii) squares are faster to render and easier to texture map (besides, we felt it was more logical to have checkered squares rather than checkered circles).

The Growing Squares visualization makes use of animation to display the dynamic execution of the system under study. While it certainly is possible to maintain all message arrows and just draw the visualization at full time, this would result in many of these messages coinciding (as in Hasse diagrams) and thus being hard to separate from other messages, as well as being impossible to associate with a specific time. Animation solves these issues in a natural way.

Another design aspect of the Growing Squares visualization is finding suitable layout methods for arranging the individual processes. Many such layout strategies exist. For instance, if the data set represents the execution of a distributed algorithm in a network, the geographical location of the individual nodes can be used to position the squares in the visualization. Other alternatives include simple grid and circular layouts (see Figure 11.3) which may serve to minimize the amount of coinciding message arrows to greater or lesser extent. In this thesis, we chose to ignore this aspect and selected a simple circular layout scheme that has the advantage of avoiding message arrows coinciding with each other or passing over processes.

11.2 User Study

Our hypothesis was that the Growing Squares technique would be faster and more efficient at quickly providing an overview of the causal relations in a distributed system than traditional methods. Furthermore, we postulated that the new technique would scale better with system size. To test these predictions, we conducted a formal comparative user study of the old Hasse diagram visualization and our new Growing Squares technique. The focus of this user study was to evaluate user performance of the "overview tasks", i.e. tasks associated with the general comprehension of how a system works. We also wanted to get a subjective assessment of the two methods.

11.2.1 Subjects

We recruited 12 subjects for this study, four of which were female. All subjects were screened to have good computer skills and basic knowledge of distributed systems and general causal relations. In particular, knowledge of Hasse diagrams was required. Subject ages ranged from 20 through 50 years old, and all had

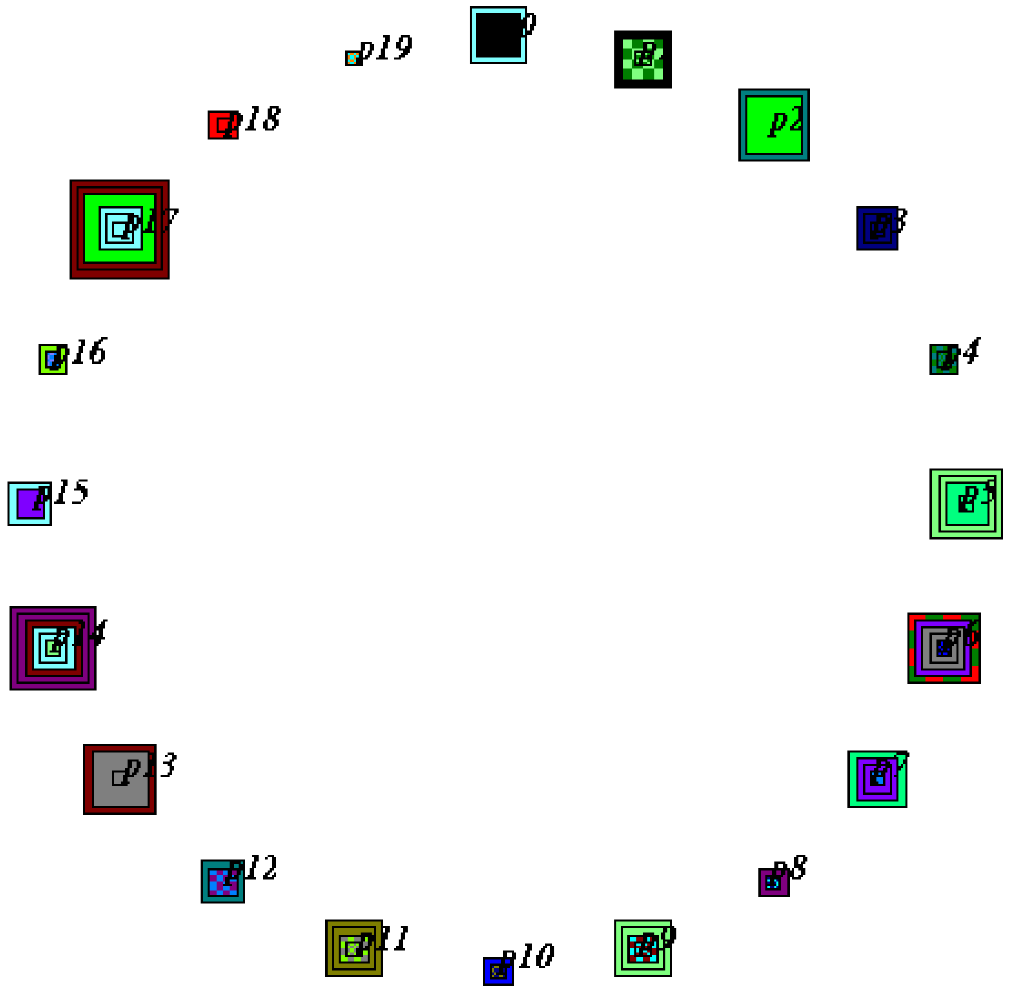


Figure 11.3: Growing Squares visualization with 20 processes.

normal or corrected-to-normal vision.

11.2.2 Equipment

The study was run on an Intel Pentium III 866 MHz laptop with 256 MB of memory and a 14-inch display. The machine was equipped with an NVidia Geforce 2 GO graphics accelerator and ran Redhat Linux 7.2.

11.2.3 Procedure

The experiment was a two-way repeated-measures analysis of variance (ANOVA) for independent variables VISUALIZATION with two levels ("Hasse diagrams" versus "Growing Squares"), and DENSITY, also with two levels. The two levels of data density were “sparse” and “dense” with 5 processes sending 15 messages and 30 processes sending 90 messages, respectively. The visualization type was a within-subjects factor, as was the data density. Each subject received the various task sets and visualization methods in counterbalanced order to avoid systematic effects of practice.

The same set of four different data sets were used for all subjects. Two were geared at the sparse case with 5 processes and 15 messages (one for each visualization type), and two for the dense case with 30 processes and 90 messages (see Table 11.1). The traces were all generated using a heuristic algorithm to avoid users taking advantage of special knowledge about real system traces. In the case of deducing inter-node causal relations, care was taken to ensure that the complexity of this was equivalent for both task sets of each density.

The evaluation procedure consisted of repeating overview tasks using Hasse diagrams and Growing Squares for first the sparse and then the dense data densities. The repeated tasks for each density and visualization type are summarized in Table 11.2. Prior to starting work on each task set, subjects were given the chance to adjust the window size and placement to their liking. Subjects were informed that they should solve the tasks quickly and focus on using the visualization to get an overview of the system trace. The completion of each task was separately timed, except for the tasks Causality 1-3, which were timed together.

We enforced an 8 minute (480 seconds) time cap on the completion of each task in order to avoid excessive times skewing the results of the user study. Uncompleted or skipped tasks were set to the time cap for that particular task.

Since we were targeting overview tasks, it was not necessary for subjects to find a precise answer to each exercise. Instead, it was deemed sufficient if subjects named one of the processes in the top 20%[2] for each category; i.e. for 30 processes, it was enough to pick one of the six processes that were most the influential, long-lived or influenced ones for the answer to be counted as correct. Only the Causality 1-3 tasks required a totally accurate answer.

After having performed each task set for a density and visualization type, subjects were asked to give a subjective rating of the efficiency, ease-of-use, and enjoyability of the visualization technique. When all of the tasks were completed, the subjects responded to a final questionnaire comparing the two visualization techniques based on the previously stated criteria (see Table 11.3).

Each evaluation session lasted approximately one hour. Subjects were given a training phase of ten minutes to familiarize themselves with the CausalViz application and the two visualization techniques. During this time, subjects were instructed in how to use the visualizations to solve various simple tasks.

[2]This number was somewhat arbitrarily chosen, partly because it was felt to be an acceptable margin of error, and partly because 20% out of 5 processes for the sparse data set translates to finding the single correct process for each task.

Data Density	**Processes**	**Messages**
Sparse	5	15
Dense	30	90

Table 11.1: Experimental design. Both density and visualization factors were within-subjects for all 12 subjects.

Task	**Description**	**Measure**
Duration	Find the process with the longest duration.	Time
Influence 1	Find the process that has had the most influence on the system.	Time
Influence 2	Find the process that has been influenced the most.	Time
Causality 1-3	Is process x causally related to process y?	Time
Q1	Rate the visualization w.r.t. ease-of-use (1=very hard, 5=very easy).	Likert
Q2	Rate the visualization w.r.t. efficiency (1=very inefficient, 5=very efficient).	Likert
Q3	Rate the visualization w.r.t. enjoyability (1=very boring, 5=very enjoyable).	Likert

Table 11.2: Repeated tasks for each density and visualization type.

11.3 Results

After having conducted the user study, we analyzed the resulting test data. The results can be divided into two parts: the objective performance measurements, and the subjective ratings of the test subjects.

11.3.1 Time

The mean times of performing a full task set (i.e. four tasks) using the Hasse diagrams and the Growing Squares visualizations were 416.58 (s.d. 268.99) and 334.79 (s.d. 230.86) seconds respectively. This, however, is not a significant difference ($F(1,11) = 2.54$, $p = .139$). The main effect for density was strongly significant ($F(1,11) = 30.99$, $p < .001$), with means for the sparse and dense conditions of 222.96 (s.d. 77.24) and 528.42 (s.d. 272.94) seconds. Figure 11.4

Task	**Description**
PQ1	Rank the visualizations w.r.t. ease of use.
PQ2	Rank the visualizations w.r.t. efficiency.
PQ3	Rank the visualizations w.r.t. enjoyability.

Table 11.3: Post-evaluation ranking questions.

summarizes the mean task results for the two visualizations across the two densities; error bars show standard deviations. The figure also shows that the mean time for the task set was higher for the Hasse method across all densities. For the sparse conditions the visualization type was significant ($F(1, 11) = 15.82$, $p = .002$), with mean values of 259.50 (s.d. 75.23) and 186.42 (s.d. 62.46) seconds for the Hasse and Growing Squares visualizations. The Growing Squares method also gave better results for dense conditions; the mean times in Hasse and Growing Squares were 573.67 (s.d. 302.96) versus 483.17 (s.d. 243.94) seconds. This, however was not a significant difference ($F(1, 11) = 1.03$, $p = .332$).

The only exception where Hasse diagrams performed better than Growing Squares is the Duration subtask for dense systems, while our technique performed better than Hasse diagrams in all other subtasks across both densities. For the Duration subtask, the mean completion times for the sparse data set using Hasse diagrams were 30.92 seconds (s.d. 9.99) versus 21.17 seconds (s.d. 17.93) for the Growing Squares method, while the mean times for the dense set were 37.00 (s.d. 15.72) and 54.75 (s.d. 28.08), respectively. This, however, was not a significant difference for this subtask ($F(1, 11) = 0.492$, $p = 0.498$). For the Influence 1 subtask the mean completion times for the sparse data set using Hasse diagrams were 75.33 seconds (s.d. 50.71) versus 65.33 seconds (s.d. 35.93) for the Growing Squares method, while the mean times for the dense set were 234.67 (s.d. 141.81) and 157.17 (s.d. 66.85), respectively. The visualization type did not have a significant effect on the completion time for this subtask ($F(1, 11) = 2.80$, $p = 0.122$). Similarly, the Influence 2 subtask yielded mean completion times of 76.17 (s.d. 31.94) versus 47.17 (s.d. 31.65) for the sparse data set, and 165.58 (s.d. 159.21) versus 136.00 (s.d. 114.83) for the dense case. Again, the type of visualization did not have a significant effect to the completion time for this subtask ($F(1, 11) = 1.062$, $p = 0.325$). Finally, the Causality 1-3 subtask resulted in sparse mean completion times of 77.08 (s.d. 31.04) for Hasse diagrams and 52.75 (s.d. 13.32) for Growing Squares, whereas the dense means were 136.42 (s.d. 88.25) and 135.25 (s.d. 77.87), respectively. The type of visualization did not have a significant effect to the completion time for this subtask ($F(1, 11) = 0.707$, $p = 0.418$).

The subjects' comments revealed that one of the reasons for the absence of a statistically significant difference between visualizations in the dense condition was because of color similarities. Much time was spent by subjects matching colors to each other and looking up process numbers in the color legend.

Subjects made little use of the animation controls in the Growing Squares visualization except to play it through once at the beginning of each task to gain a picture of the data set. Only a few of the subjects actively moved the timeline back and forth to solve various subtasks, and most preferred to leave the time setting at the end of the execution.

The fixed (circular) layout algorithm used in the user study turned out to be limiting when it came to comparing the size (i.e. duration) of individual processes. Users remarked that it would have been useful to be able to click and drag processes to arbitrary positions to facilitate comparison as well as to group processes into semantic clusters (i.e. clusters of the same perceived type).

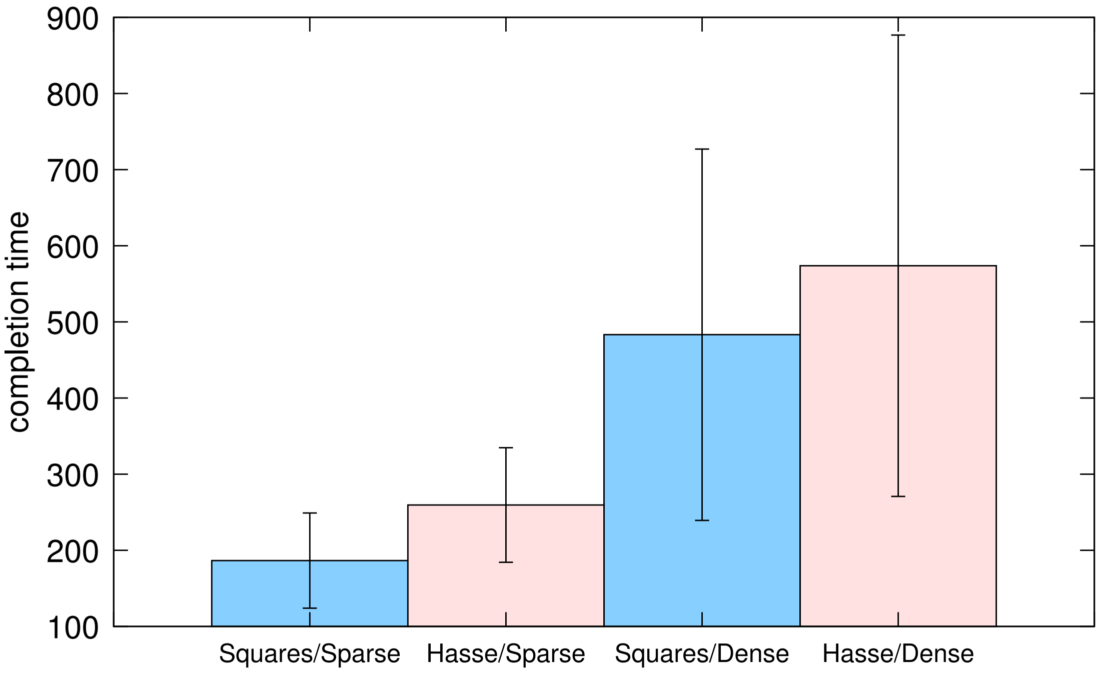


Figure 11.4: Mean task completion times for all tasks across the Hasse and Growing Squares methods and across levels of density. Error bars show standard deviations.

11.3.2 Subjective Ratings

The subjects consistently rated Growing Squares above Hasse diagram with respect to efficiency, ease-of-use and enjoyment. The mean response values to the five-point Likert-scale questions are summarized in Figure 11.5. The complete data analysis table is presented as Table 11.4.

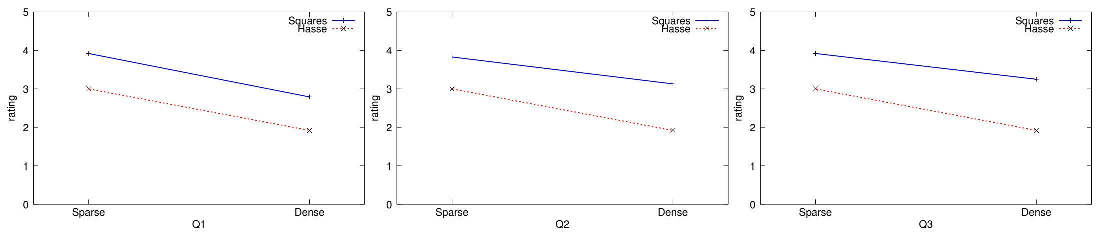


Figure 11.5: Responses to Q1-Q3 5-point Likert-scale questions across sparse and dense data densities for the Hasse and Growing Squares methods.

The subjects' responses to the efficiency question (Q2, Table 11.4) showed a higher rating for the Growing Squares visualization than Hasse diagrams in both sparse (means 3.83 (s.d. .39) and 2.75 (s.d. .97)) and dense data densities (means 3.13 (s.d. .68) and 1.58 (s.d. .67)). Both higher rating readings were significant (Friedman Tests, $p = .0209$ for the sparse case and $p = .0039$ for the dense case). The subjects' response to the ease-of-use question (Q1, Table 11.4) also showed a higher rating for the Squares visualization in both sparse (means 3.92 (s.d. .67) and 2.67 (s.d. .89)) and dense data densities (means 2.79 (s.d. .78) and 1.46 (s.d. .66)). Both higher rating readings were significant (Friedman Tests, $p = .0094$ for the sparse case and $p = .0015$ for the dense case). The subjects' response to the enjoyment question (Q3, Table 11.4) also showed a higher rating for the Squares visualization in both sparse (means 3.92 (s.d. .79) and 3.00 (s.d. .43)), and dense data densities (means 3.25 (s.d. .85) and 1.92 (s.d. .67)). Both higher rating readings were significant (Friedman Tests, $p = .0094$ for the sparse case and $p = .0015$ for the dense case).

Figure 11.5 shows, not surprisingly, that the density of the data set strongly influenced the subjects' responses to each question for both visualizations. This difference is reliable for all but the enjoyability question (Friedman Tests). The subjects' response to this question (Q2, Table 11.4) when using the Growing Squares visualization shows a higher rating when small data sets are considered (means 3.92 (s.d. .79) for sparse sets and 3.25 (s.d. .75) for large sets), but on the other hand, this is not a significant difference ($p > .05$).

The final ranking questionnaire shows that most subjects preferred the Growing Squares technique over Hasse diagrams with regard to ease of use, efficiency, and enjoyment (Table 11.5). Overall, the results from this ranking are very favorable for the Growing Squares method.

11.4 Caveats of Growing Squares

The Growing Squares technique is based on animation, colors and patterns to improve the perception of causality in distributed systems, and the results from

Question	**Hasse diagrams**		**Growing Squares**	
	sparse	**dense**	**sparse**	**dense**
Q1. Ease-of-use rating	2.67 (.89)	1.46 (.66)	3.92 (.67)	2.79 (.78)
Q2. Efficiency rating	2.75 (.97)	1.58 (.67)	3.83 (.39)	3.13 (.68)
Q3. Enjoyability rating	3.00 (.43)	1.92 (.67)	3.92 (.79)	3.25 (.75)

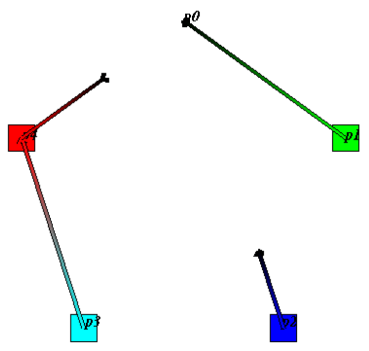
Table 11.4: Mean (standard deviation) responses to 5-point Likert-scale questions.

	Question	**Prefer GS?**
PQ1	Rank visualizations w.r.t. ease-of-use.	92%
PQ2	Rank visualizations w.r.t. efficiency.	83%
PQ3	Rank visualizations w.r.t. enjoyability.	92%

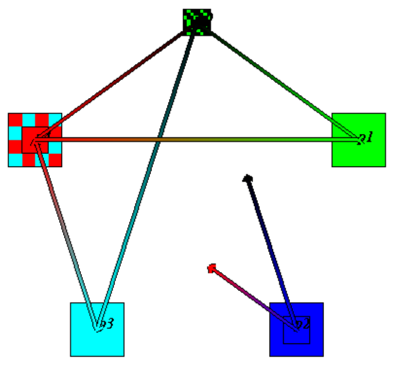
Table 11.5: Subject responses to ranking the two visualizations.

the user study show that the technique is consistently faster and more efficient than Hasse diagrams. This difference, however, is not statistically significant for the general case, although it is significant for the sparse data set case. While there clearly is room for additional improvement, the Growing Squares visualization technique is nevertheless an improvement over conventional Hasse diagrams.

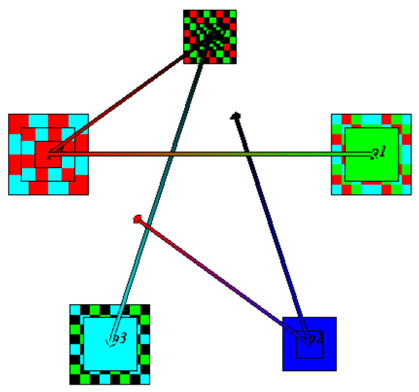
However, as indicated by the user study, the Growing Squares technique has a number of issues. First, since the method is dependent on a simple color coding for each process in a system, it is often very difficult to distinguish individual processes in a large system due to the similarity of the colors. This problem is exacerbated by the fact that Growing Squares presents the influences of a single process as colored pixels in a checkered pattern on each square, meaning that each influence can become arbitrarily small due to limited screen space (this problem is partially solved using a continuous zoom mechanism, however). Finally, a Growing Squares visualization does not explicitly communicate the absolute timing of events or process startup or shutdown; this must be manually deduced by studying the animated execution of the system.



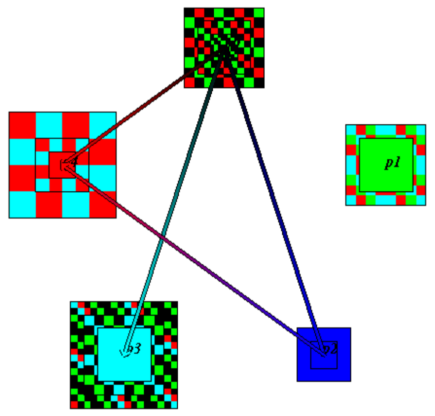
(a)



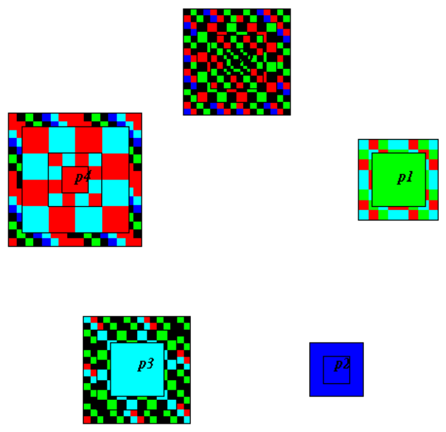
(b)



(c)



(d)



(e)

Figure 11.6: Growing Squares visualization of the dynamic execution of a 5-process distributed system.

Chapter 12

Growing Polygons

Visualizing the causal relations in a system consisting of n processes using the Growing Polygons [45] technique is done by placing n-sided polygons (so-called *process polygons*) representing the individual processes on the sides of a large n-sided polygon (the *layout polygon*). Instead of using a linear timeline, as in Hasse diagrams, the time parameter is mapped to the size of each process polygon so that they grow from zero to maximum size as time proceeds from the start to the end of the execution, just like in the Growing Squares technique. The visualization is animated to allow the user to study the dynamics of the execution, and the discrete time steps are shown as dashed "age rings" in the interior of each polygon. In addition to this, each process polygon is divided into triangular sections, with every process in the system being assigned a color and a specific sector in the polygon. This sector also corresponds to the side where the process polygon is positioned on the layout polygon. Whenever the process represented by a particular polygon is active, the appropriate time segments of the associated sector in the polygon will be filled in with the process color. Messages between processes in the system are shown as arrows traveling from the source polygon to the destination, and will activate the corresponding sector in the destination polygon with the color of the source process. In other words, a message sent from process A to process B will contaminate A's sector in B starting from the time the message was received.

Figure 12.4 shows an example of a simple 3-process system (consisting of processes P_0, P_1, and P_2) where each process is represented by a triangle partitioned into three sections, and with the process triangles positioned on the sides of a larger layout triangle. For each process triangle, the process's own sector has been marked with a thick black outline, and the internals of each polygon has been segmented to show the discrete time steps of the execution. In addition, the processes have been assigned the colors red, green, and blue, respectively. In this example, we see how P_0 sends a message to P_1 at $t = 0$ that reaches the destination process at time $t = 1$, establishing a causal relation between the two nodes. Notice how for all times $t \geq 1$, P_0's sector within P_1's process triangle is now filled, signifying this influence. By studying the polygons at $t = t_{end}$, i.e. the end of the execution, we can get a clear picture of the flow of information within the system.

As we ascertained earlier, causal relations are transitive, so if $A \rightarrow B$ and $B \rightarrow C$, then $A \rightarrow C$. Figure 12.4 also shows how this is expressed in the Growing Polygons visualization. At time $t = 2$, process P_2 receives a message from P_1. P_1 has already been influenced by P_0 in the previous interaction (in other words, there is already a causal relation between P_0 and P_1). Thus, the process triangle of P_2 now shows causal influences in *all* of its process sectors, including the transitive dependency to P_0, not just the direct dependency to P_1, which sent the actual message.

The simple execution in Figure 12.4 also gives information about the absolute lifecycles of the three processes. By studying the filled segments of each process triangle's own sector, we note that only process P_0 executed from the start to the end of the system trace; processes P_1 and P_2 were kickstarted by external messages at times $t = 1$ and $t = 2$, respectively. In fact, unlike the Growing Squares technique, the new method allows users to deduce the exact timing of all events in a system since the age rings in the interior of each polygon are fixed to absolute times.

Just like the Growing Squares technique, the Growing Polygons technique offers a view of the transitive closure of the node dependencies and influences, facilitating analysis of global information flow in the system (and not just locally, as for Hasse diagrams). The visualization is animated and can thus avoid many of the message intersection problems of Hasse diagrams. In addition to this, by assigning not only a color but also a fixed polygon sector to each process, the Growing Polygons method largely remedies the difficulty of distinguishing colors that plague the Growing Squares technique. Thus, the new method is considerably more scalable than the old one since it is now enough that two similar colors are not placed in adjacent sectors for a user to be able to separate them.

Now let us study a full example to see the Growing Polygons visualization in action. Figure 12.5 shows a sequence of screenshots taken at the discrete time steps of the execution of a 5-process system (in the real visualization, these images are smoothly animated). The processes are laid out in clockwise order with P_0 at the top right. In (a), at $t = 1$, we see that all processes except P_0 are executing and sending messages (the process sector of P_0 is empty). However, a message from P_1 is just about to reach P_0 and will activate it starting from this point in time. Screenshot (b) shows the subsequent situation at $t = 2$, where P_0 now has begun executing and exhibits a causal dependence to the green process (P_1) that started it, and where P_4 similarly shows a dependence to P_3 (P_3's sector in P_4's process polygon is filled in from time step 1 and onwards). Moving to $t = 3$ in (c), we see more causal dependencies appearing in the process polygons of the various nodes, the transitive dependencies in both P_1 (cyan from P_3) and

P_3 (green from P_1) being of special interest. We can also observe that process P_2 appears to have stopped executing since it is no longer filling up its own process sector. Image (d) displays the situation one time step later ($t = 4$), where the two messages from the inactive P_2 finally reach P_0 and P_4 respectively, and image (e) shows the final situation at $t = 5$, with the causal dependencies in the system plainly visible.

12.1 Design

One of the weaknesses of the original Growing Squares method that limited its scalability was the difficulty of distinguishing between different process colors. To remedy this problem, the Growing Polygons technique also assigns a unique triangular sector to each process. Nevertheless, for our method to work efficiently, adjacent process sectors should not have similar colors, or users can easily mistake one process for another. Just like in the Growing Squares case, we opted for a straightforward non-continuous distribution of colors across the RGB spectrum.

While our new method does not exhibit the same congestion of screen space that plagues Growing Squares, where a much-influenced process square simply cannot convey all of its influences in its limited screen space, there are instances where even Growing Polygons fail at this. For example, when visualizing a large system with many processes, the angle ($\theta = 360°/n$) assigned to each process sector will be small, making it difficult to distinguish events early on in the execution. The same is also true if the time span of the execution is long, since the layout algorithm will then have to scale each time step to fit inside the allocated maximum size of each polygon. To cope with these two situations, the Growing Polygons visualization retains the simple continuous zoom mechanism of the Growing Squares technique, allowing users to zoom in arbitrarily close in order to distinguish details in the visualization.

The decision to use animation in the Growing Polygons technique was mainly grounded on the wish to avoid a maze of cris-crossing message arrows (like in Hasse diagrams). At the end of the system execution, no message arrows at all are visible, facilitating easy study of the inter-process dependencies in the system. Animation allows the user to still see the dynamic execution of the system in an intuitive way, just like in the Growing Squares technique.

12.2 User Study

Our intention with the Growing Polygons technique was to provide an efficient way of viewing the flow of information and the node dependencies in a system of communicating processes. In order to check whether our method performs better than existing methods, we conducted a comparative user study between

Hasse diagrams and Growing Polygons. The study involved test subjects that were deemed representative of the target audience, and consisted of having them solve problems using the two techniques. Time performance and correctness were measured, as well as the subjective ratings of individual users.

12.2.1 Subjects

We recruited 20 subjects for this study, five of which were female. All subjects were screened to have good computer skills and at least basic knowledge of distributed systems and general causal relations. Subject ages ranged from 20 through 50 years old, and all had normal or corrected-to-normal vision (one person claimed partial color blindness but was still able to carry out the test). Ten of the subjects had participated in our earlier user study of the Growing Squares technique.

12.2.2 Equipment

We used the same equipment that was used for the Growing Squares user study for this study as well (see Section 11.2.2).

12.2.3 Procedure

As before, the experiment was a two-way repeated-measures analysis of variance (ANOVA) for the independent variables VISUALIZATION ("Hasse diagrams" versus "Growing Polygons") and DENSITY ("sparse" versus "dense"). The sparse data density consisted of system executions involving 5 processes and 15 messages, while the dense data density involved 20 processes and 60 messages (see Table 12.1). All subjects were given the same four task sets split into the two density classes. The system trace for each task set was generated using a simple randomized heuristic algorithm to avoid subjects taking advantage of special knowledge about the behavior of a particular distributed system. In addition, care was taken to ensure that the complexity of both system traces for a specific data density was roughly equivalent by removing ambiguities and ensuring that the number of indirect relations was the same.

The procedure consisted of solving two of the four task sets using conventional Hasse diagrams, and the other two using the Growing Polygons technique. Sparse task sets were solved first, followed by the respective dense sets. In order to minimize the impact of learning effects, half of the subjects used the Hasse diagrams first, while the other half used the Growing Polygons first. The task sets themselves consisted of four tasks that were directly based on our previous user study of Growing Squares (see Table 11.2 for an overview). Subjects were given the opportunity to adjust window size and placement freely prior to starting work on each task set. Furthermore, subjects were instructed to solve each task quickly but thoroughly, and were allowed to ask questions during the course of the procedure. Each individual task in a task set was timed separately, except for the tasks Causality 1-3, which were timed together. In addition, answers were checked and the correctness ratio was recorded for each task.

In order to avoid run-away times on troublesome tasks, completion times were limited to 10 minutes (600 seconds). If a test subject chose for some reason to skip a task, the completion time for that task was set to this cap.

After each completed task set, each subject was given a short questionnaire of three 5-point Likert-scale questions asking for their personal opinion on the usability, efficiency, and enjoyability of the visualization method they had just used (see tasks Q1 to Q3 in Table 11.2). The purpose of this questionnaire was to measure how users' ratings of the visualizations changed depending on the data density. In addition, users also filled out a post-evaluation questionnaire after having completed all of the task sets, where they were asked to rank the two visualizations on the above criteria (see Table 12.2).

Each evaluation session lasted approximately 45 minutes. Prior to starting the evaluation itself, subjects were given a training phase of up to ten minutes where they were given instructions on how to use both visualization methods to solve various simple tasks.

Data Density	**Processes**	**Messages**
Sparse	5	15
Dense	20	60

Table 12.1: Experimental design. Both density and visualization factors were within subjects for all 20 subjects.

Task	**Description**
PQ1	Rank the visualizations w.r.t. ease of use.
PQ2	Rank the visualizations w.r.t. efficiency for solving the following tasks: (a) Duration analysis (b) Influence importance (most influential) (c) Influence assessment (most influenced) (d) Inter-node causal relations
PQ3	Rank the visualizations w.r.t. enjoyability.

Table 12.2: Post-evaluation ranking questions.

12.3 Results

The analysis of the results we obtained from the user study can be divided into the timing performance, the correctness, and the subjective ratings of the test subjects.

12.3.1 Time

The mean times of solving a full task set (i.e. all four tasks) using Hasse diagrams and the Growing Polygons visualizations were 433.90 (s.d. 378.59) and 251.85 (s.d. 174.88) seconds respectively. This is also a statistically significant difference ($F(1, 19) = 20.118$, $p < .001$). The main effect for density was significant ($F(1, 19) = 26.932$, $p < .001$), with means for the sparse and dense conditions of 191.80 (s.d. 87.57) and 493.95 (s.d. 359.35) seconds.

Figure 12.1 summarizes the mean task results for the two visualizations across the two densities; error bars show the standard deviation above and below the mean. The figure also shows that the mean time for the task set was higher for the Hasse method across all densities. For the sparse condition, the mean completion times were 234.40 (s.d. 87.09) and 149.20 (s.d. 65.85) seconds for the Hasse and Growing Polygons visualizations. The Growing Polygons method also gave better results for dense conditions, with mean values of 616.05 (s.d. 550.60) seconds for the Hasse visualization versus 354.50 (s.d. 190.41) seconds for Growing Polygons.

The one exception where Hasse diagrams performed better than Growing Polygons was for the Duration subtask across both densities, with sparse set mean times of 25.75 (s.d. 10.39) for Hasse diagrams versus 33.95 (s.d. 17.47) for Growing Polygons, and for the dense set, 34.40 (s.d. 18.54) versus 72.35 (s.d. 36.06) seconds. This difference was also significant ($F(1, 19) = 26.943$, $p < .001$).

For the Influence 1 subtask, on the other hand, the mean completion times for the sparse data set using Hasse diagrams was 58.50 seconds (s.d. 22.25) versus 36.65 seconds (s.d. 17.93) for the Growing Polygons method, while the mean times for the dense set were 270.60 (s.d. 180) and 169.70 (s.d. 140.72), respectively. This was a significant difference ($F(1, 19) = 14.614$, $p = 0.001$). Similarly, the Influence 2 subtask yielded mean completion times of 77.64 (s.d. 53.58) versus 34.35 (s.d. 30.47) for the sparse data set, and 184.10 (s.d. 207.05) versus 50.85 (s.d. 26.61) for the dense case. Again, this was a significant difference in favor of the Growing Polygons method ($F(1, 19) = 14.170$, $p = 0.001$). Finally, the Causality 1-3 subtask resulted in sparse mean completion times of 72.50 (s.d. 29.28) for Hasse diagrams and 44.25 (s.d. 19.68) for Growing Polygons, whereas the dense means were 144.30 (s.d. 116.37) and 61.60 (s.d. 40.88), respectively. This was also a significant difference ($F(1, 19) = 18.896$, $p < 0.001$).

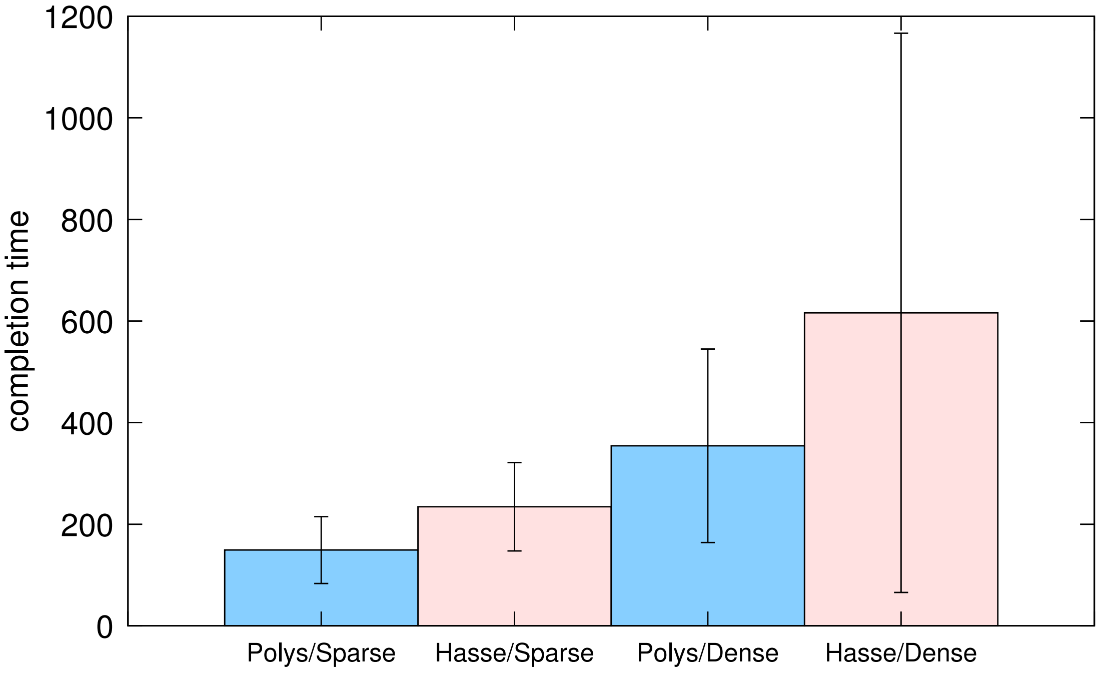


Figure 12.1: Mean task completion times for all tasks across the Hasse and Growing Polygons methods and across levels of density. Error bars show standard deviations.

12.3.2 Correctness

The average number of correct answers when solving a full task set (i.e. six tasks) using Hasse diagrams and the Growing Polygons visualization was 4.375 (s.d. 1.148) versus 5.625 (s.d. 0.667) correct answers, respectively. This is a significant difference ($F(1, 19) = 46.57$, $p < .001$). For the sparse data set, the mean correctness was 4.70 (s.d. 1.218) for Hasse diagrams and 5.75 (s.d. 0.716) for Growing Polygons, versus 4.05 (s.d. 0.999) and 5.50 (s.d. 0.607) for the dense case. In fact, the mean correctness of the Growing Polygons visualization is significantly better than for Hasse diagrams for all individual subtasks except for the Duration subtask, where Hasse performs better with a correctness ratio of 0.975 versus 0.950 for Growing Polygons. This, however, is not a significant difference ($F(1, 19) = 0.322$, $p = .577$). See Figure 12.2 for a diagram of the mean correctness values for all tasks.

12.3.3 Subjective Ratings

For the post-task questionnaire, the test subjects consistently rated Growing Polygons above Hasse diagram in all regards, including efficiency, ease-of-use and enjoyment. The mean response values to the five-point Likert-scale questions are

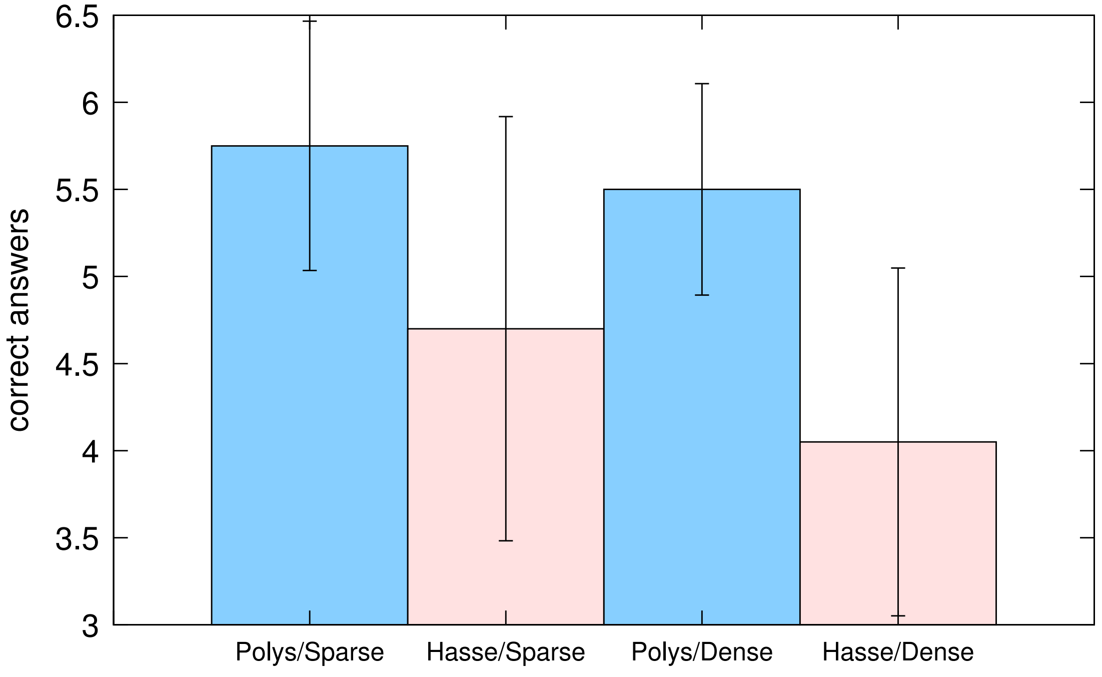


Figure 12.2: Mean correctness for all tasks across the Hasse and Growing Polygons methods and across levels of density. Error bars show standard deviations.

summarized in Figure 12.3. See Table 12.3 for the complete data analysis table.

The subjects' response to the ease-of-use question (Q1, Table 12.3) showed a higher rating for the Growing Polygons visualization than Hasse diagrams in both sparse (means 4.20 (s.d. .70) and 2.75 (s.d. .85), respectively) and dense data densities (means 3.75 (s.d. .79) and 1.90 (s.d. .91)). Both higher ratings were significant (Friedman Tests, $p < .001$ for both the sparse and dense cases). The subjects' responses to the efficiency question (Q2, Table 12.3) showed a higher rating for the Growing Polygons visualization in both sparse (means 4.20 (s.d. .62) and 2.40 (s.d. .88)) and dense data densities (means 3.95 (s.d. .51) and 1.55 (s.d. .51)). Both higher ratings readings were significant (Friedman Tests, $p < .001$ for the sparse case and $p < .001$ for the dense case). The subjects' response to the enjoyment question (Q3, Table 12.3) also showed a higher rating for the Growing Polygons visualization in both sparse (means 4.20 (s.d. .62) and 2.95 (s.d. .39)), and dense data densities (means 4.10 (s.d. .64) and 2.00 (s.d. .73)). Both higher ratings were significant (Friedman Tests, $p < .001$ for the sparse case and $p < .001$ for the dense case).

The results from the post-task summary questionnaire can been found in Table 12.4, and clearly show that test subjects regard the Growing Polygons technique as superior to Hasse diagrams in all aspects except for duration analysis (task PQ2 (a)). However, as can be seen from the table, the overall user rankings are very convincingly in favor of our method.

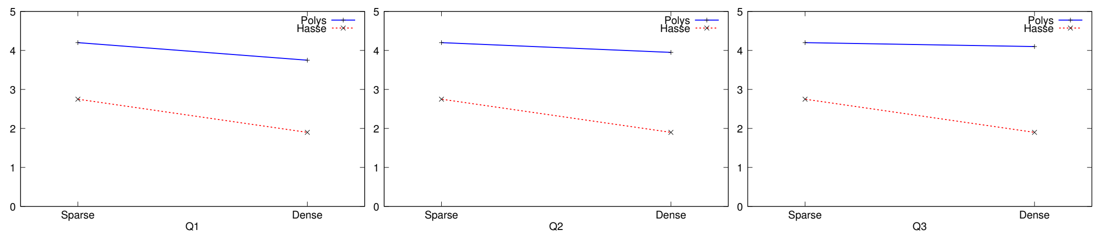


Figure 12.3: Responses to Q1-Q3 5-point Likert-scale questions across sparse and dense data densities for the Hasse and Growing Polygons methods.

Question	**Hasse diagrams**		**Growing Polygons**	
	sparse	**dense**	**sparse**	**dense**
Q1. Ease-of-use rating	2.75 (.85)	1.90 (.91)	4.20 (.70)	3.75 (.79)
Q2. Efficiency rating	2.40 (.88)	1.55 (.51)	4.20 (.62)	3.95 (.51)
Q3. Enjoyability rating	2.95 (.39)	2.00 (.73)	4.20 (.62)	4.10 (.64)

Table 12.3: Mean (standard deviation) responses to 5-point Likert-scale questions.

Task	**Description**	**GP**	**Hasse**	**Undecided**
PQ1	Ease-of-use	95%	0%	5%
PQ2	Efficiency (avg)	80%	11%	9%
	(a) Duration	35%	40%	25%
	(b) Importance	90%	5%	5%
	(c) Assessment	95%	0%	5%
	(d) Causality	100%	0%	0%
PQ3	Enjoyability	100%	0%	0%

Table 12.4: Subject responses to ranking the two visualizations.

12.4 Caveats of Growing Polygons

While the Growing Polygons technique is clearly an improvement over both the classic Hasse diagram as well as our earlier alternative causality visualization technique, Growing Squares, there is certainly room for improvement. Specifically, for very large systems of causal relations with large numbers of processes spanning a long period of time, the Growing Polygons technique may perform poorly, simply due to the limited screen estate available to display a potentially huge amount of data. In order for our technique to be able to handle these extremes, we have to consider implementing mechanisms for space distortion and hierarchical node clustering into the technique.

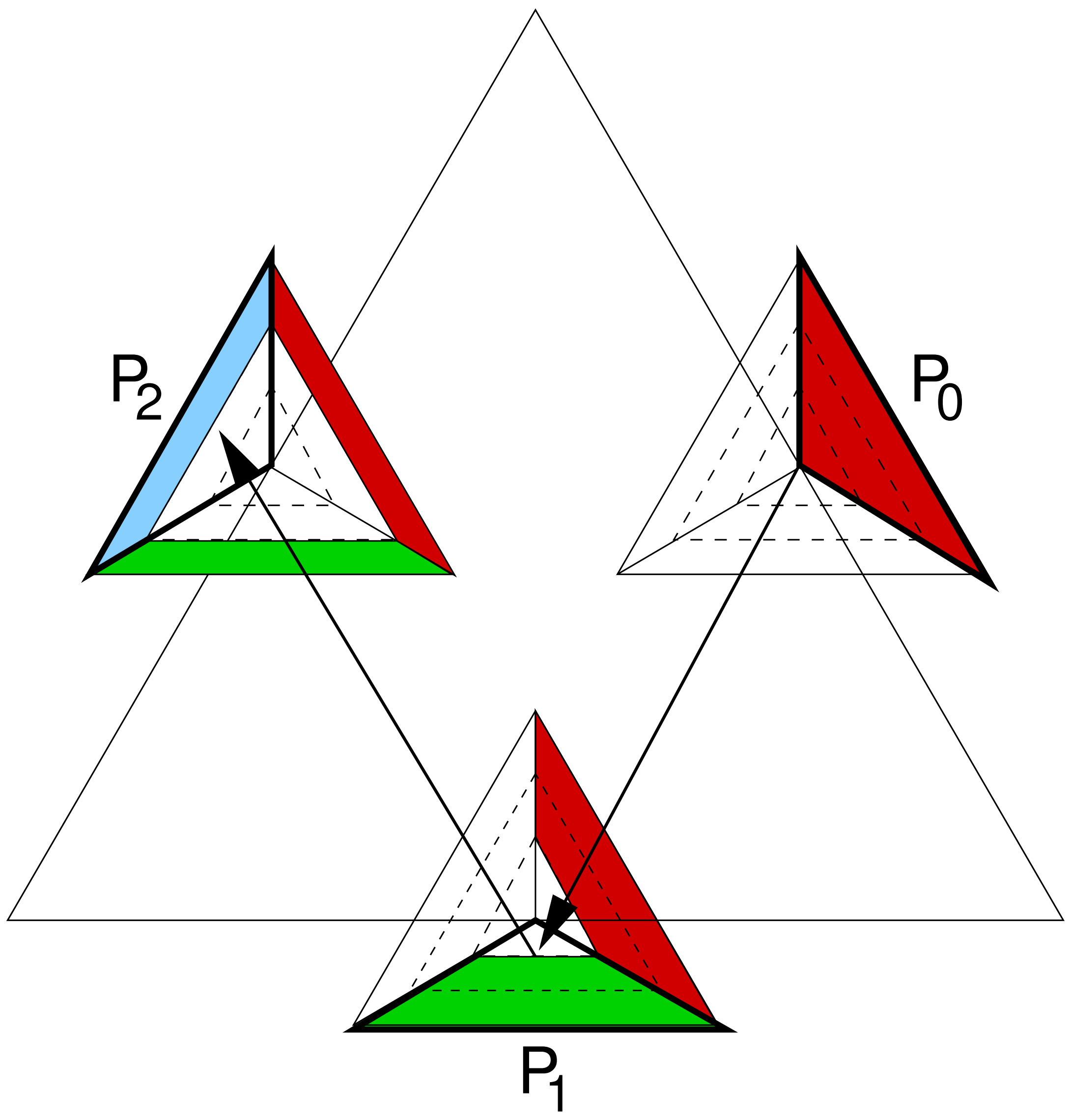


Figure 12.4: Growing Polygons visualization with $n = 3$ (i.e. the process polygons are triangles).

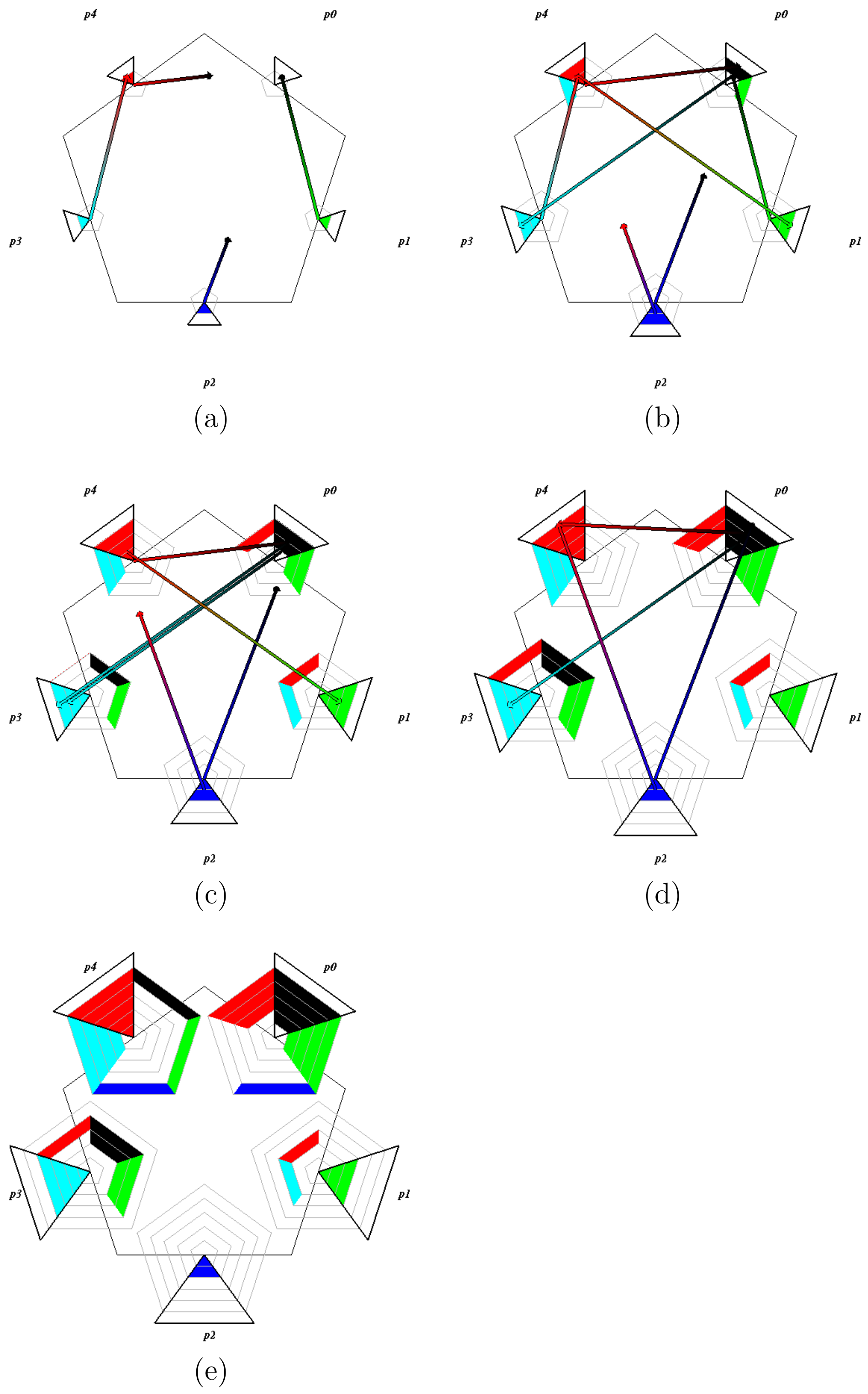


Figure 12.5: Growing Polygons visualization of the dynamic execution of a 5-process distributed system.

Chapter 13

Citation Network Visualization

One of the key tasks of scientific research is the management and study of existing work in a given field of inquiry. The specific nature of the tasks involved in this venture vary greatly depending on the situation and the role of the researcher; for a new student just entering a research area, the task is that of orientation within the existing work; for a reviewer, one of originality and correctness checking; for a conference organizer, one of chronological survey; and, finally, for an experienced scientist, one of staying abreast with new developments and identifying current hot topics in his or her area of choice. Researchers spend a considerable portion of their time on these tasks, ample evidence that it is in everyone's best interest to streamline this process as much as possible, and that large time savings can likely be made.

In this chapter, we present CiteWiz, a tool for bibliographic visualization of the chronology and influences in networks of scientific articles. The tool contains three primary visualizations: a timeline visualization for overviews and navigation in a full citation database, an influence visualization for detailed views of a specific subset of the citation database, and an interactive concept map for exploring keywords and co-authorship in the database. Users would typically orient themselves in a citation database using the timeline and the concept map and then construct specialized database views to study using the influence visualization. The tool was designed for use by researchers, scientists, and students alike, and its baseline features were established through extended discussions in a focus group consisting of such users. These discussions helped us formulate a taxonomy for citation database interaction that we believe is a valuable contribution to the area. Guided by this taxonomy and the focus group, we created a prototype implementation of the tool with a user interface that allows for normal browsing and filtering of the citation meta-data as well as building hierarchical views of the dataset for visualization. We have conducted a formal user study to assess the efficiency of the tool in comparison with standard web-based database interfaces. Our results indicate that CiteWiz is equally efficient as standard database interfaces for low-level analysis tasks such as finding papers and correlating authors, and significantly more efficient than standard databases for higher-level analysis tasks related to overviews and influences of bibliographical data.

Causality and influences both play a large role in tracing the history of ideas and trends in a scientific community and these are core strengths of the Growing Polygons [45] technique. In order to allow us to make use of this technique, we show how to model citations in scientific articles using general causal relations, and we introduce the slightly relaxed concept of *influence* between articles in a citation database. We chose an article-centered approach (as opposed to an author-centered one) in our implementation, where the articles themselves are the active entities (represented by processes), and citations are the information-bearing messages between them. To allow the technique to cope with potentially huge datasets, we also improved its scalability in two different ways: we implemented multi-level process hierarchies for grouping sets of articles together, and we added a focus+context technique with variable time scale to handle long event histories. The visualization was accordingly supplemented with a number of interaction techniques to support these new features as well as techniques targeted specifically at citation visualization; these include collapsing and expanding the group hierarchy, navigating in the citation network by following backward and forward references, and getting details-on-demand of the complete bibliographical data for a specific paper.

13.1 Related Work

The common model of viewing citation networks as directed graphs (see the next section) lends itself quite naturally to visualizing bibliographical data as simple node-link diagrams. However, node-link diagrams scale poorly with network size and only present local dependency information: it is easy to see direct citing and cited articles, but the user must traverse the graph in order to see dependencies more than one step away. CiteWiz, on the other hand, provides the surrounding context through influence mapping, and gives a more straightforward way to see the chronology of articles.

Modjeska et al. [104] propose a minimum set of functions necessary for effective bibliographic visualization: (i) display of complete bibliographic information, (ii) filtering by record fields, (iii) display of chronology and influence of articles, (iv) information views at different levels of detail, (v) multiple simultaneous views, and (vi) visualization of large search results. They also present the BIVTECI prototype system that partially implements this specification, but the visualization used in the tool is restricted to node-link diagrams with visualized attributes. CiteWiz also implements this minimum functionality, but instead employs the Growing Polygons causality visualization technique in order to handle larger search results and provide stronger chronology information.

The RefViz [118] system is a commercial visualization tool for scientific citation networks. It is based on two main views: a “galaxy” network view showing clusters of papers based on their conceptual relations, and a concept occurrence “matrix” view showing the distribution of concepts being discussed in relation to each other. This is orthogonal from the CiteWiz approach, where the basis for the visualization is the actual references in each paper.

The Butterfly [99] system provides a 3D visualization front-end of the DIALOG science citation databases, using the notion of “organic user interfaces” to build an information landscape as the user explores the results of various queries. Individual articles are represented by an innovative butterfly-shaped 3D object with references and citers on the left and right wings, respectively, and various graphical cues are provided to orient the user when browsing the citation network. Butterfly uses a node-link diagram for overview and context, however, and has no mechanism for showing the cumulative influences and chronology of articles.

The results from the InfoVis 2004 contest [55] are of special interest to this work, especially since we use the same citation database for our prototype implementation. Many excellent entries, developed concurrently with our work, were presented at the contest. Ke et al. [83] use a graph visualizer to show the relationship between the major papers in the database, scaling node size proportionally to the number of citations just like entities in the CiteWiz timeline visualization. Delest et al. [38] use a related approach. Keim et al. [84] employ InterRings [147] to visualize co-authorship, a technique somewhat similar to using Growing Polygons for co-authorship studies.

CiteWiz and the above-mentioned systems are all article-focused tools in that they emphasize the visualization of articles and their interdependencies. A number of group-focused techniques have also been proposed, where the emphasis lies on representing the groupings and structure of a scientific domain through metrics such as relevance, bibliographic coupling [85], and co-citation [127]. Work in this area is numerous but peripheral to the system described in this paper; examples include [19, 25, 27, 28, 75].

13.2 Citation Networks

Citation networks consist of bibliographical entries representing scientific works, each being a tuple of attributes such as title, authors, source, date, abstract, keywords, etc. In addition, each entry has a number of references to other entries representing the citations found in the article. Thus, citation networks can be seen as directed graphs where each node represents an article, out edges represent cited papers (i.e. the dependencies of the current paper), and in edges represent citing papers. A citation graph is generally not acyclic since articles may mutually cite each other; this is often the case when an author (or a team of authors) publishes two or more related articles to the same conference or journal issue.

Traditional bibliographical databases generally provide means for searching, sorting, and filtering the citation data in various ways (examples include IEEE Xplore (*http://ieeexplore.ieee.org/*), the ACM Digital Library [39], and CiteSeer [63]). These database interfaces serve as suitable reference implementations when assessing new visualizations for citation networks.

13.2.1 Formative Evaluation

In order to deduce the common user tasks associated with bibliographical databases, we organized a formative user evaluation using a focus group of six active researchers from our department. Our intention with this session was to identify the high-level issues and tasks involved with the use of bibliographical data, including various situations when researchers make use of such databases. This session influenced the development of our taxonomy of citation database interaction based on user roles and the tasks and subtasks associated with each role. This taxonomy, presented in the following section, has proven useful when discussing bibliographic visualization and the analysis tasks involved in this activity, but may have a slight bias towards a researcher's point of view; we plan to involve other users of citation databases (e.g. librarians) in future updates of the taxonomy.

13.2.2 Taxonomy of Citation Database Interaction

A researcher may assume any of a number of different roles when interacting with a citation database, and we have thus chosen to base our taxonomy on the concept of *user roles* and the goals and tasks associated with these. Clearly, a user has different *goals* to achieve depending on his or her current role, and these govern which *tasks* need to be performed. Using this taxonomy, we can make decisions about which user roles and goals we want a tool to support, and accordingly which tasks we must implement.

In the taxonomy below, the terms *group* and *subgroup* refer to any (potentially hierarchical) clustering of articles (and subgroups) according to some criteria, such as shared keywords, author, source, etc. An *event* is defined as any scientific community activity, such as a journal issue, a conference, a workshop, etc. Furthermore, we have categorized the user tasks depending on the focus of the task; making a distinction between (i) article-, (ii) event-, (iii) author-, and (iv) group-focused user tasks is useful when discussing the nature of a visualization tool.

Table 13.1 presents the roles we have identified, including a short description of each role. Table 13.2 gives a listing of the individual goals of each role, as well as the tasks involved with completing that particular goal. Finally, Table 13.3 shows the different tasks, including their focus category. Note that these tasks operate on the current working group and not necessarily the entire database; for instance, task T3 should be interpreted as "find the most influential paper in the current group of papers."

13.2.3 Citations as Causal Relations

A causal ordering is a general relation that relates two *events* where one is the cause of the other. We can interpret citations in scientific articles as causal orderings in at least two different ways: either with authors as the active entities (processes) and their papers as events, or with papers as the active entities and a single event marking the paper's publication for each entity. For both cases, we represent citations by causal relations between the events. In this work, we have chosen the latter approach for the simple reason that the former causes problem with the visualization when authors combine to work together on a paper; thus, our visualization is fundamentally article-focused instead of author-focused.

Seeing that a citation in a scientific article can be modeled by a causal relation is straightforward: a citation implies that (a) the authors have read and somehow been influenced by the cited paper (and thus, indirectly, that the cited paper existed before the citing paper), and that (b) the citing paper has a dependency to the cited paper. Admittedly, mutual citations cannot be represented and must be either removed entirely or broken arbitrarily. However, these occur very seldom in practice, so this is a minor point. In this paper, we will use the term *influence*, which is a relaxed interpretation of causality in this context: if a paper A cites a paper B, the authors of A have been influenced (in some undefined way) by paper B, and this is reflected in the paper (put shortly, A has been influenced by B).

13.3 The CiteWiz Platform

The CiteWiz system is a modularized bibliographic visualization platform based on a central citation dataset and a number of dataset views that can be used as input for the available visualization techniques. The three primary visualizations in CiteWiz include a timeline visualization and an interactive concept for overview, and an influence visualization for detail views. The interactive concept map visualization allows for exploring keywords and co-authorship in the citation database. Users typically employ the timeline visualization and the concept map to orient themselves in a dataset and then study subsets of the dataset views using the Growing Polygons method. Based on the taxonomy described earlier, we developed the tool to be primarily article-focused, meaning that we emphasize the visualization of articles and their interdependencies, but sufficient provisions exist for author-, group-, and event-focused user tasks as well.

Role	**Description**
Novice	A researcher that is new to a specific field; can either be a new student or an experienced researcher moving to a new area.
Expert	An experienced researcher with intimate knowledge of a field.
Reviewer	A researcher tasked with peer-reviewing a new paper, potentially from a field of which he or she has only passing knowledge.
Organizer	A researcher responsible for organizing, editing, and/or steering an event (such as a conference or journal).
Evaluator	A person, such as a recruiter, tasked with evaluating the work of a specific researcher.

Table 13.1: User roles in citation database usage.

13.3.1 Datasets and Views

CiteWiz has a central citation dataset that is used for all queries and visualizations. Each entry in the set is a name/value pair, with fields for the conventional attributes such as title, authors, source, keywords, abstract, etc. Entries also have a list of references to other entries cited in the paper. The dataset is loaded from disk using a simple XML-based file format for citation meta-data that was designed for the InfoVis 2004 contest [55].

Users can browse, filter, sort, and search the dataset in the CiteWiz application. In addition, users can also build *views* of the dataset for visualization; these are essentially subsets of the central dataset with the extra capability of containing hierarchical groups of bibliographical entries. This makes it possible to build complex structures of nested groups according to some criteria relevant to the user. For instance, when studying a dataset containing citation data for a specific conference over a period of time, one might create groups for each conference year, and the papers could then be arranged in subgroups representing the different sessions for each conference. Other groupings are possible and depend on the user's goals. For instance, when performing author-focused tasks, it might be useful to create groups for each author in the dataset and add their papers, allowing for easy study of author chronology and collaboration.

Views can be saved and loaded to disk using another straightforward XML format; each view file is associated with a specific dataset file, and uses the internal identifiers to refer to bibliographical entries in the dataset.

Role	**Goal**	**Tasks**
Novice	Orientation in a new area	T2, T3, T5, T6
	Find open problems	T4
Expert	Verify hypotheses/intuition	T1
	Stay updated	T1
	Find papers quickly	T1
Reviewer	Check originality	T2, T3, T5
	Check correctness	T2, T3
	Check adequacy of references	T2, T5
Organizer	Identify hot topics	T4, T5, T6
	View chronology of an event	T7
	View collaborations between events	T8
Evaluator	View the career of an author	T7
	Assess the work of an author	T2, T3, T5

Table 13.2: Goals for each user role.

13.3.2 Timeline Visualization

The overview visualization of the CiteWiz tool is informally referred to as a Newton's Shoulders diagram.[1] This visualization creates a timeline of either articles or authors in the central CiteWiz citation database, displaying each entity as an icon on the timeline according to their publication date (or the date of their first publication, in the case of authors). The user task we want to support is overview, specifically the tasks T3, T7 and T8 in our taxonomy.

The surface area of each icon is scaled proportionally to the amount of citations the article or author has received (rounded up so that the icon conforms to a uniform grid). The timeline is split into suitable time units (years or months), and each time segment is assigned space on the timeline equal to the size of the largest entity in the segment. The icons representing the entities for each time segment are then laid out using a greedy algorithm that places the entities in descending size within the allocated space on the timeline, always trying to minimize the distance to the center point of the diagram. An example of such a Newton's Shoulders diagram can be seen in Figure 13.1 depicting a modest-sized citation database of some 1000 authors. Our implementation allows the user to zoom and pan smoothly in the visualization.

[1]So named after Sir Isaac Newton's famous quote in a letter to Robert Hooke in 1676, "If I have seen further, it is by standing on the shoulders of giants."

Task	**Description**	**Focus**
T1	Find a particular paper	Article
T2	Find related papers	Article
T3	Find the most influential paper(s)	Article
T4	Find hot topics (at a specific time)	Group
T5	Partition an area into subareas	Group
T6	Study the overall citation network	Article
T7	Study the chronology of an author/event/group	Au/Ev/Gr
T8	Study the collaboration between authors/events/groups	Au/Ev/Gr

Table 13.3: Tasks for citation database interaction.

Furthermore, we can orient the timeline vertically and use human figures for the entity icons, giving the impression of people standing on the shoulders of others. This is exactly the metaphor we had in mind when designing the visualization, and matches the intuition of the work of a researcher resting on the work of those who came before. The diagram now tells us the relative chronology of researchers in a specific field, and instantly shows the most influential authors and their relationships.

These diagrams can be modified to show additional dimensions by applying color to the entity icons. The choice of metric to display this way can be chosen arbitrarily; one ad-hoc metric for authors could be *citation density*, which we define as the total number of citations for an author divided by the total number of publications written by the author (i.e. a kind of "average paper quality" metric). Another, slightly more complex, metric would involve weighing citations for an author or article by their age so that recently cited articles or authors receive a stronger and more visible color than older ones, signifying that this article or author is involved in a "hot topic".

13.3.3 Interactive Concept Map

To further support orientation in a citation database, the CiteWiz platform also includes an interactive concept map visualization based on a basic force-directed graph layout scheme [40]. The purpose of this visualization is to give users an overview of the keywords and co-authorship in the database, but additional concept graphs can also be visualized, including author influences and article authorship (i.e. connect articles with at least one shared co-author). The user task we are addressing here is again overview, specifically the tasks T3, T4, and T8 in our taxonomy.

The concept map visualization is implemented as a simple graph where each concept is mapped to a node and associations between concepts are undirected edges. For purposes of physical simulation, nodes have an associated weight

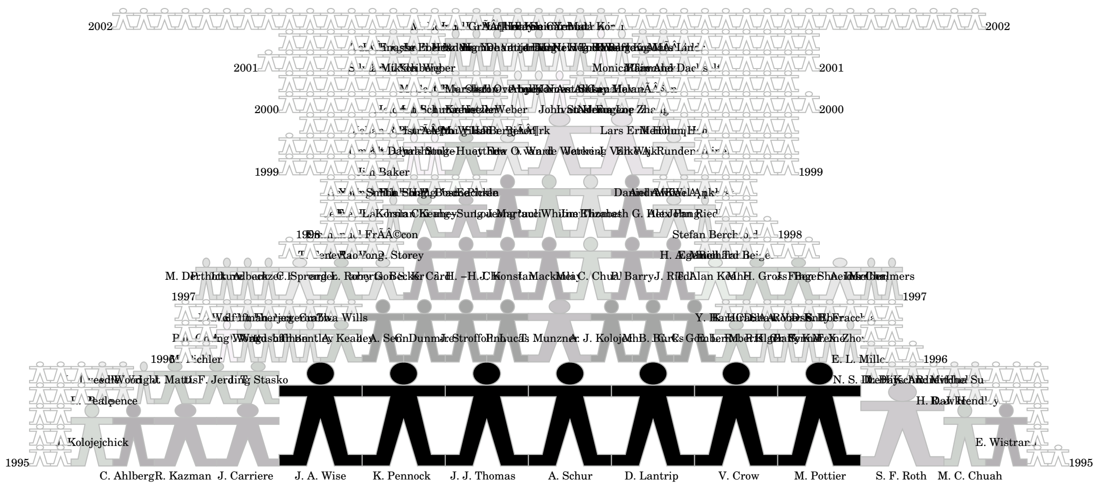


Figure 13.1: Timeline visualization of InfoVis authors.

(which is used to derive the node size), and edges are modeled as linear springs with a specific spring constant and a normal length. We employ Hooke's Law to derive the force exerted on two nodes connected by a spring, and sum these up to form the resultant force. Furthermore, we also connect each node to the center of the canvas using an invisible "leash" to force nodes to gravitate towards the center of the visualization. Both of these are modulated by the node mass. Dampers are associated with each node to help the system reach a stable state. Finally, we use Newton's Law of Gravity to generate repelling forces between nodes to avoid them overlapping each other.

In our implementation, we perform real-time numerical simulation of the physical system to show the result. Users can interact with individual nodes by clicking and dragging on them, causing a spring to be temporarily connected between the mouse pointer and the node. The user can also control the dampening, repelling, spring and leash constants for the visualization.

We construct various kinds of concept maps depending on the user task. For keyword maps, we traverse the database and add all keywords as nodes, using their individual frequency as mass. Links are created between any two keywords that appear in the same article. See Figure 13.2 for a keyword concept map of the IV04 contest dataset. We can plainly see the main themes of the citation database, i.e. information visualization, data visualization, visualization, and so on.

For co-authorship maps, on the other hand, we add authors as nodes, using their number of citations as mass, and connect each pair of authors with a link for every article they have co-authored. See Figure 13.3 for a co-authorship map for the same dataset. Again, this visualization allows us to quickly pinpoint the main contributors to the citation database and the clusters they form.

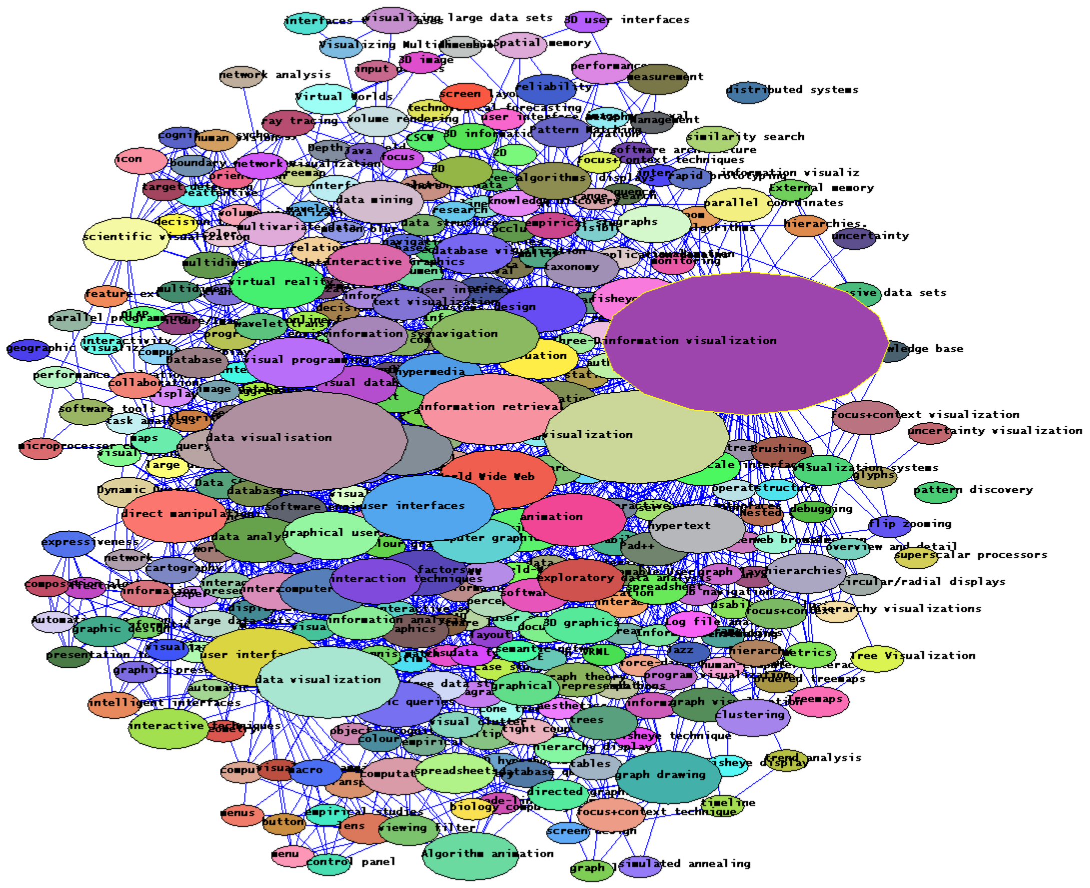


Figure 13.2: Concept map for the keywords in the InfoVis dataset.

13.3.4 Influence Visualization

After having studied the general shape of a citation database using the Newton's Shoulder visualization, users are able to build views tailored to answer their specific queries about the dataset. These views form the input for the Growing Polygons [45] method for visualization of general causal relations, suitably modified to be able to handle citation networks and the scalability issues associated with these. We believe the focus on influence and causality visualization in the Growing Polygons technique makes it very well suited to visual exploration of citation networks. The technique uses a combination of 2D shapes, color, and animation to represent a system of n active processes graphically as n-sided so-called *process polygons* showing the influences affecting each process. As time progresses, the process polygons grow from zero to full size, and the sectors of each polygon fill in as messages are received by other processes, signifying the information transfer.

In our adaptation of the original technique, articles form the processes in the

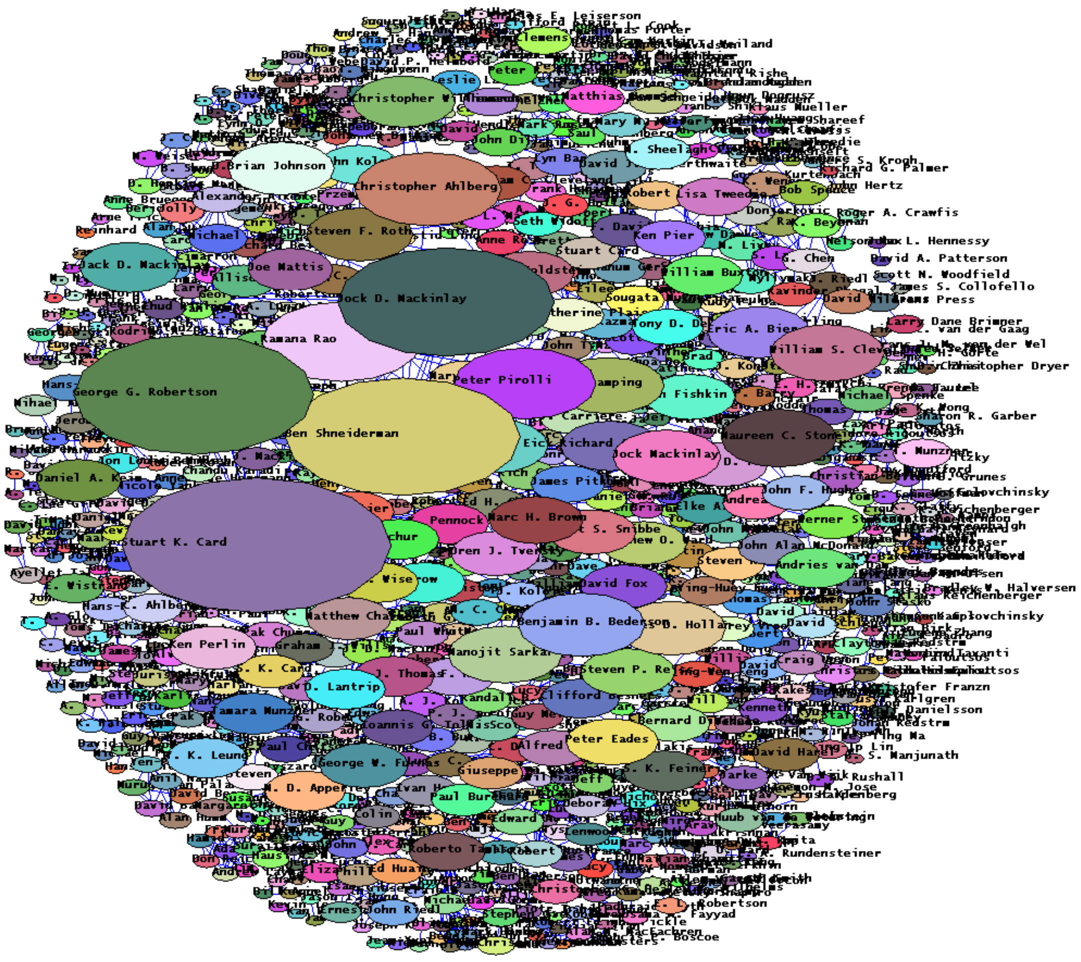


Figure 13.3: Concept map for co-authorship in the InfoVis dataset.

visualization (thus represented by *article polygons*), and citations are messages from a source (cited) article to a destination (citing) article. This mimics the information transfer implicit when authors reference another paper. Even if articles are more or less static once published, this article-focused approach gives us a way to see the influences and chronology of a set of articles easily, including global transitivity information for each article. The user tasks in the taxonomy we address are primarily T2, T3, T4, and T6.

In order to make effective use of the Growing Polygons method in this context, we addressed two scalability issues in relation to (i) long execution times, and (ii) large quantities of visualized articles. For the former issue concerning time scalability, the problem lies in that visualizing a large citation network may result in very long chains of causality, and the visualization will then run out of space for displaying individual time segments. For the latter case, the quantity scalability issue comes from the fact that visualizing a sufficiently large amount of articles means that each individual article is assigned a very small polygon sector and it will thus be difficult to distinguish between neighboring sectors. Both of these issues can be partially addressed through zooming mechanisms, but this instead results in loss of overview.

Our solution for these concerns in the modified, more scalable version of the Growing Polygons method is two-fold. First, we introduce a focus+context [59] technique based on adjustable *linear time windows* that lets the user concentrate on certain areas of the execution while still retaining the context of the surrounding history (i.e. the focus view and the overview are integrated in the same visual space, as opposed to overview+detail techniques where the views are spatially or temporally separate). Second, we address the quantity concern by modifying the Growing Polygon technique to handle *hierarchical views* instead of flat article lists (this was our incentive for the distinction between datasets and views in the design of CiteWiz). Note that the purpose of the influence visualization is not for overview of huge datasets, but for detailed studies of customized views (on the order of a hundred documents).

Hierarchical Views

In order to allow the Growing Polygons technique to handle a large quantity of articles, we modify the visualization to be able to render hierarchical groups of articles instead of single articles. These correspond directly to the views of the central dataset built by the users. The view hierarchy is visualized by treating an article group as a normal article, except that the group will have the cumulative influences of all of its children. We derive these influences by a simple postorder traversal of the hierarchy, building the influence timelines of the internal nodes from the bottom up (i.e. starting with the articles in the leaves of the tree). The currently visible nodes are rendered as normal article polygons, with the single exception that groups (i.e. non-leaves) have a drop shadow to signify that the polygon represents more than one article.

Interaction Techniques

In our modified version of the Growing Polygons technique, we provide two simple interaction techniques for browsing and exploring the article hierarchy: users can either click directly in the visualization to expand and collapse article groups (using the left and right mouse buttons, respectively), or they can use a separate tree navigation window to study the structure of the hierarchy. The same tree window can also be used to search for the full or partial name of a specific article, and the tree will be expanded to the level of the article to show the search result.

In addition to these techniques, we also provide an overview map window with a color legend and clickable fields for quickly jumping to a specific article

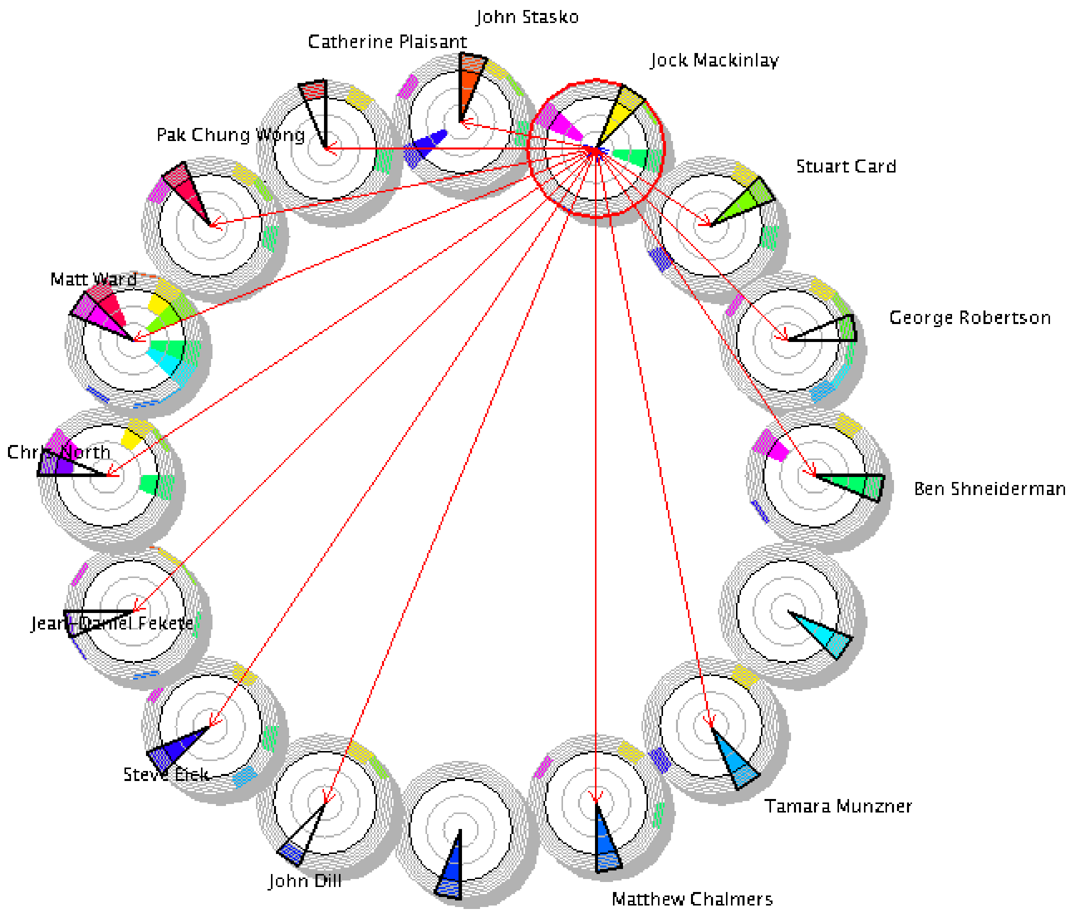


Figure 13.4: Visualization of selected InfoVis authors.

polygon.

Details-On-Demand

As suggested by both Shneiderman [123] and Modjeska et al. [104], bibliographic visualization tools need to provide a mechanism to show the complete bibliographical data of an article. In CiteWiz, this is handled by a detail window that gives the full meta-data of the currently selected article. Users can easily navigate through the references of articles, as well as moving back and forth in the article history of the window. In addition to this, we augment the currently selected node in the visualization with blue arrows pointing from the cited nodes, and with red arrows pointing to citing nodes (see Figure 13.4)—again for convenience reasons.

13.3.5 Implementation

The CiteWiz tool is implemented as a C++ application running under the Linux operating system, but should be easily portable to other platforms. It uses standard OpenGL for efficient 2D rendering, and the GTK+/GTK– library for the graphical user interface components.

13.4 User Study

The purpose of the CiteWiz citation visualizer was to provide researchers with additional tools for analyzing citation data beyond the standard low-level features available in most traditional database interfaces. We chose the IEEE Xplore database as a good baseline database web interface for comparison. Our hypothesis was that CiteWiz would perform as well as IEEE Xplore for finding papers and correlating bibliographical data, and that CiteWiz would perform significantly better for higher-level tasks.

13.4.1 Subjects

We recruited 10 test subjects for this study, one of which was female. All subjects were active researchers at our department, but were carefully screened to have no previous knowledge of the IEEE InfoVis citation database. Furthermore, all subjects had considerable previous experience in the use of citation database interfaces. Ages ranged from 25 to 40. All subjects had normal or corrected-to-normal vision.

13.4.2 Equipment

The study was run on an Intel Pentium III 1 GHz desktop computer with 512 MB of memory and a 19-inch color display. The machine was equipped with an NVidia Geforce 3 graphics card and ran Redhat Linux 9.

13.4.3 Procedure

We designed the test to be a between-subjects comparative study of a traditional database interface versus our CiteWiz citation visualizer. We selected three different tasks related to citation database interaction from our taxonomy presented earlier in this paper: T1, T3, and T8 (see Table 11.2). We designed the scenarios for T1 and T8 to consist of low-level analysis tasks such as searching, filtering, and correlating basic bibliographical data; the scenario for T3, on the other hand, required a higher-level analysis of influence and structure of the citation network.

Completion time was capped at 15 minutes to avoid runaway tasks; subjects were given the option to abandon a troublesome task, in which case the completion time was set to the cap. Subjects were allowed a 5-minute training period prior to performing the test for both tools, and were asked to fill out a questionnaire after having completed it.

We selected the IEEE Xplore web-based database interface as a suitable representative of traditional database interfaces. IEEE Xplore is widely used among scientists all over the world to access the bibliographical data and fulltexts of IEEE publications and supports all standard search and filtering features. Accordingly, we selected all of the papers of the IEEE InfoVis conferences from 1995 to 2002 as our test database (175 articles). Albeit a small dataset, this was a necessary delimitation for us to be able to use the same database for both tools. In order to remove all distractions, we were able to design our own search interface to the IEEE Xplore database (essentially a cleaner version of the standard IEEE Xplore Basic Search), allowing us to constrain searches to the InfoVis conference and provide a browseable list of the InfoVis proceedings sorted by year. The CiteWiz XML-based database, on the other hand, was adapted from a subset of the InfoVis 2004 contest database [55].

13.5 Results

The main findings of the user study were the expected ones: that (i) there is no significant difference in efficiency for CiteWiz and IEEE Xplore for simple tasks involving finding papers and collating basic citation data, and that (ii) CiteWiz is significantly more efficient for a higher-level task involving the study of dependencies and influences of a set of articles.

13.5.1 Time

The mean times of solving a full task set (i.e. all three tasks) using IEEE Xplore and CiteWiz were 20 minutes and 2 seconds (s.d. 158 seconds) and 8 minutes 5 seconds (s.d. 72 seconds), respectively. This was a statistically significant difference ($t(8) = -9.19, p < 0.000$). For task T1, the mean completion times were 3 minutes 15 seconds for IEEE Xplore versus 3 minutes 20 seconds for CiteWiz. No user managed to solve task T3 within the 900 second time cap using IEEE Xplore (two subjects completed the task, three abandoned the task); the CiteWiz completion time was 2 minutes 34 seconds. Finally, for task T8, the mean completion times were 1 minute 46 seconds versus 2 minutes 20 seconds for IEEE Xplore and CiteWiz, respectively.

	Question	Xplore	CiteWiz
Q1.	Ease-of-use	3.40 (1.67)	4.20 (1.10)
Q2.	Efficiency		
	(a) Find paper	4.40 (.55)	4.20 (.84)
	(b) Most influential	1.00 (.00)	4.60 (.55)
	(c) Collaboration	2.40 (.89)	3.20 (1.10)
Q3.	Enjoyability	2.60 (.89)	4.20 (.84)

Table 13.4: Mean (standard deviation) responses to 5-point Likert-scale questions.

13.5.2 Correctness

No subject using the IEEE Xplore managed to solve task T3 correctly (even when exceeding the time cap), while all subjects using CiteWiz correctly solved T3. All subjects registered correct answers on all other tasks.

13.5.3 Subjective Ratings

The ratings from the post-test questionnaire overall show encouraging results; see Table 13.4 for an overview. Note especially the responses to question Q2b and Q2c, which show subjective ratings strongly in favor of CiteWiz over IEEE Xplore. In general, users consistently perceived the CiteWiz tool as more enjoyable to use than IEEE Xplore.

13.6 Discussion

Our expectations of the results of the user study was that CiteWiz and the IEEE Xplore tool would perform equally well at low-level tasks related to basic searching, sorting and correlation of bibliographical data, and that higher-level tasks involving assessing influences and structure of the citation network would yield a significantly higher efficiency for CiteWiz. These expectations were fulfilled. It is certainly possible to improve standard database interfaces with better support for these higher-level analysis tasks, but this is not always practical; for instance, the IEEE publications database does not contain reference information (the ACM Digital Library does, however). Nevertheless, the purpose of this work was mainly to target the deficiencies of existing standard tools, and in this regard we succeeded.

The visual browsing and exploration features that CiteWiz provide are very hard to measure qualitatively in comparison to standard databases, but the test subjects expressed great enthusiasm when exposed to this visualization and some were very eager to use the tool in their own area of research. In fact, CiteWiz itself was used for researching previous work for this paper, and revealed a few interesting articles we had not considered.

As a final note, the task set in the user study is limited, but this was a deliberate design decision due to the small feature set that the IEEE Xplore database provides. Choosing a more complex task set would give an unfair advantage to our tool, and would punish the test subjects who used IEEE Xplore. We believe that the user study shows that there is room for improvement, and that the techniques presented in this paper are viable alternatives to traditional database interfaces.

13.7 Conclusions

We have described CiteWiz, a platform for bibliographic visualization. The platform includes a timeline visualization and an interactive concept map for overview, and a modified version of the Growing Polygons method for detailed studies. The tool is based on a taxonomy of the usage of citation databases. The timeline visualization, informally called a Newton's Shoulders diagram, constructs timelines of articles or authors showing the causality and citations in a citation database. The concept map uses a spring-directed layout to make interrelations between concepts and keywords in the database visible. The modifications to the Growing Polygons technique were aimed primarily at adapting the method to citation networks, and included provisions for rendering hierarchies of articles rather than flat lists, and a focus+context technique with user-controlled time windows to more easily support long citation chains. Finally, we presented the formal user study we have conducted, a between-subjects comparative analysis of CiteWiz in relation to the standard IEEE Xplore web-based database interface. Our results confirm our intuition: that CiteWiz and IEEE Xplore perform equally well for low-level citation interaction tasks such as correlating bibliographical data, and that CiteWiz is significantly more efficient to use for higher-level tasks such as influence and citation structure assessment.

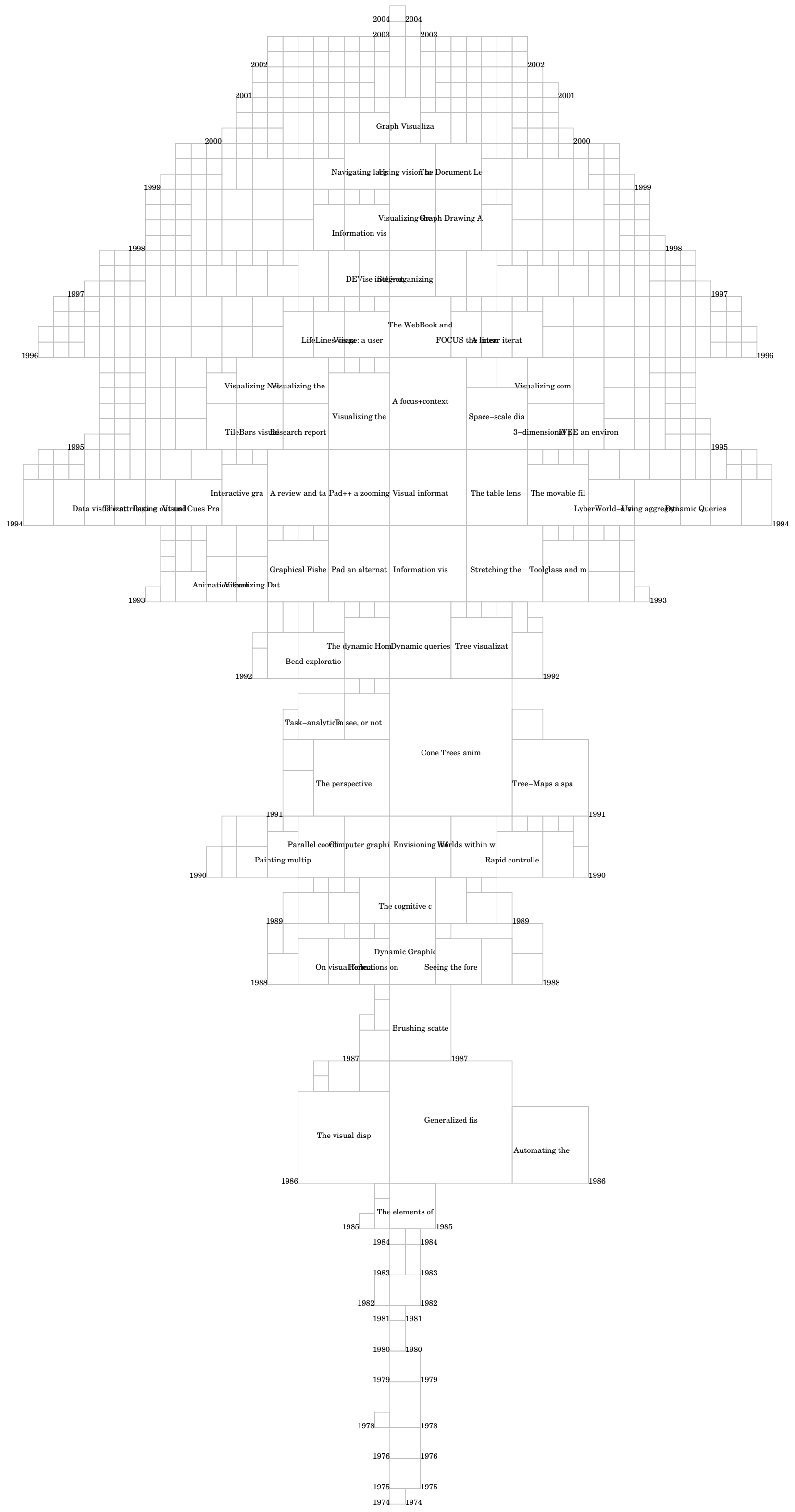


Figure 13.5: Newton's Shoulders diagram of the articles in the IV04 contest citation database.

Chapter 14

Conclusions and Future Work

Causality is central to human thinking, but the sheer complexity of most sequences of events appearing both in nature as well as in the world of computing makes overview and insight extremely difficult. The goal of this work has been to bring the methods of information visualization to bear on this problem in order to provide means with which users can study these complex causal relationships and draw conclusions from the data. In particular, we have done so by introducing two different visualization techniques for graphically representing causality on a computer. Furthermore, we have conducted a case study of visualizing citation networks—an instance of the causality visualization problem—and presented an adaptation of one of our techniques for solving it.

The results obtained from our user studies show that both the Growing Squares and the Growing Polygons methods are superior to Hasse diagrams in terms of performance, correctness, and the subjective opinion of the test subjects across all data densities (although Growing Squares are only significantly more efficient to use for the sparse density). The test subjects consistently ranked both techniques before Hasse diagrams in almost all aspects. Our findings show that users are significantly more efficient and correct when using Growing Polygons to analyze the influences and check inter-process causal relations in a system.

One interesting question is naturally where the Growing Polygons and Growing Squares techniques stand in relation to each other. While we have not performed a direct comparison between the two techniques, the Growing Polygons method is likely superior to the Growing Squares method. First of all, the Growing Polygons method achieves statistically significant improvement over Hasse diagrams in *all* subtasks (except the duration analysis subtask, which the Growing Squares method also failed at) and across all densities, something which the Growing Squares method does not manage for dense data sets. Second, the comments from the test subjects who also participated in the previous user study clearly indicate that the new method is significantly superior to the older one. Unfortunately, the nature of the work we conducted means that we cannot compare the two techniques directly.

The positive feedback that we have received from our test subjects suggests that these kinds of alternate visualization methods of causal relations are indeed useful and worthwhile avenues for future research. By combining them with traditional methods such as Hasse diagrams, users are able to harness the strengths of different methods to solve different problems. In addition, the ability to view systems of causal relations from different perspectives will greatly aid in understanding the mechanics of such a system.

Causality visualization as a whole is an interesting subfield to information visualization and has great potential for many different application areas, especially within software visualization (our original point of interest). The work presented in this thesis merely scratches the surface of this area, and much work remains to be done. More specifically, we are interested in investigating additional ways to improve the scalability properties of our visualizations, primarily when it comes to managing high numbers of processes. We are convinced that the current parent-child hierarchy visualization can be made more effective, perhaps by making use of animation to show group expansion and collapse. In addition, the improved Growing Polygons technique makes poor use of screen space, and we suspect that clever distortion of the spatial substrate, perhaps through fisheye [59] techniques, would remedy this problem.

On a more general note, visualizing general causality, or indeed something as concrete as the execution of a program (especially a distributed one), is truly a core information visualization problem in that the data is abstract and lacks a natural visual mapping. Thus, the choice of visual representation is entirely up to the designer. In this thesis, we have presented two visualization methods for this problem and motivated our choices with both basic theories taken from information visualization as well as empirical experiments, but there naturally exists a nearly unlimited number of alternative representations, many of which are bound to be superior in at least a few regards. No single visualization will ever be able to capture all the information a user might want to see, but by combining several different visualizations, we will be able to satisfy user needs much better.

Bibliography

[1] Maneesh Agrawala, Denis Zorin, and Tamara Munzner. Artistic multiprojection rendering. In *Proceedings of the Eurographics Workshop on Rendering Techniques 2000*, pages 125–136, 2000.

[2] Tomas Akenine-Möller. Fast 3D triangle-box overlap testing. *Journal of Graphics Tools*, 6(1):29–33, 2001.

[3] Christopher Alexander, Sara Ishikawa, Murray Silverstein, Max Jacobson, Ingrid Fiksdahl-King, and Shlomo Angel. *A Pattern Language*. Oxford University Press, 1977.

[4] Carlos Andújar, Pere-Pau Vázquez, and Marta Fairén. Way-finder: guided tours through complex walkthrough models. In *Proceedings of Eurographics 2004*, pages 499–508, 2004.

[5] Daniel Asimov. The grand tour: A tool for viewing multidimensional data. *SIAM Journal on Scientific and Statistical Computing*, 6(1):128–143, January 1985.

[6] Ulf Assarsson, Niklas Elmqvist, and Philippas Tsigas. Image-space dynamic transparency for improved 3D object discovery. Technical Report 2006-10, Chalmers University of Technology, 2006.

[7] Michelle Q. Wang Baldonado, Allison Woodruff, and Allan Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of the ACM Conference on Advanced Visual Interfaces 2000*, pages 110–119, 2000.

[8] Patrick Baudisch and Carl Gutwin. Multiblending: displaying overlapping windows simultaneously without the drawbacks of alpha blending. In *Proceedings of the ACM CHI 2004 Conference on Human Factors in Computing Systems*, pages 367–374, 2004.

[9] Benjamin B. Bederson, James D. Hollan, Ken Perlin, Jonathan Meyer, D. Bacon, and George W. Furnas. Pad++: A zoomable graphical sketchpad for exploring alternate interface physics. *Journal of Visual Languages and Computing*, 7:3–31, 1996.

[10] Benjamin B. Bederson, Jon Meyer, and Lance Good. Jazz: An extensible zoomable user interface graphics toolkit in Java. In *Proceedings of the ACM Symposium on User Interface Software and Technology 2000*, pages 171–180, 2000.

[11] Thomas Bemmerl and Peter Braum. Visualization of message passing parallel programs with the TOPSYS parallel programming environment. *Journal of Parallel and Distributed Computing*, 18(2):118–128, June 1993.

[12] Jacques Bertin. *Graphics and Graphic Information Processing*. De Gruyter, Berlin, 1981.

[13] Eric Bier, Maureen Stone, Ken Fishkin, William Buxton, and Thomas Baudel. A taxonomy of see-through tools. In *Proceedings of the ACM CHI'94 Conference on Human Factors in Computing Systems*, pages 358–364, 1994.

[14] Eric A. Bier, Maureen C. Stone, Ken Pier, William Buxton, and Tony DeRose. Toolglass and Magic Lenses: The see-through interface. In *Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 73–80, 1993.

[15] Blender, September 2006. See *http://www.blender3d.org*.

[16] Doug A. Bowman and Larry F. Hodges. Formalizing the design, evaluation, and application of interaction techniques for immersive virtual environments. *Journal of Visual Languages and Computing*, 10(1):37–53, 1999.

[17] Doug A. Bowman, David Koller, and Larry F. Hodges. Travel in immersive virtual environments: An evaluation of viewpoint motion control techniques. *Proceedings of the IEEE Conference on Virtual Reality 1997*, pages 45–52, 1997.

[18] Doug A. Bowman, Chris North, Jian Chen, Nicholas F. Polys, Pardha S. Pyla, and Umur Yilmaz. Information-rich virtual environments: theory, tools, and research agenda. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology 2003*, pages 81–90, 2003.

[19] Ulrik Brandes and Thomas Willhal. Visualization of bibliographic networks with a reshaped landscape metaphor. In *Proceedings of the Eurographics Symposium on Data Visualisation 2002*, pages 159–164, 2002.

[20] Stuart Card, Jock Mackinlay, and George Robertson. The design space of input devices. In *Proceedings of the ACM CHI'90 Conference on Human Factors in Computing Systems*, pages 117–124, 1990.

[21] Stuart K. Card and Jock Mackinlay. The structure of the information visualization design space. In *Proceedings of the IEEE Symposium on Information Visualization 1997*, pages 92–99, 1997.

[22] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman, editors. *Readings in information visualization: Using vision to think*. Morgan Kaufmann Publishers, San Francisco, 1999.

[23] Ingrid Carlbom and Joseph Paciorek. Planar geometric projections and viewing transformations. *ACM Computing Surveys*, 10(4):465–502, December 1978.

[24] Loren Carpenter. The A-buffer, an antialiased hidden surface method. *Computer Graphics*, 18(3):103–108, July 1984.

[25] Matthew Chalmers and Paul Chitson. Bead: Explorations in information visualization. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval 1992*, pages 330–337, 1992.

[26] William G. Chase. Visual information processing. In Kenneth R. Boff, Lloyd Kaufman, and James P. Thomas, editors, *Handbook of Perception and Human Performance, Vol II: Cognitive Processes and Performance*, pages 28–1–28–71. John Wiley and Sons, 1986.

[27] Chaomei Chen. Visualising semantic spaces and author co-citation networks in digital libraries. *Information Processing and Management*, 35(3):401–420, 1999.

[28] Chaomei Chen and Steven Morris. Visualizing evolving networks: Minimum spanning trees versus pathfinder networks. In *Proceedings of the IEEE Symposium on Information Visualization 2003*, pages 67–74, 2003.

[29] Luca Chittaro and Stefano Burigat. 3D location-pointing as a navigation aid in virtual environments. In *Proceedings of ACM Conference on Advanced Visual Interfaces 2004*, pages 267–274, 2004.

[30] Luca Chittaro, Roberto Ranon, and Lucio Ieronutti. Guiding visitors of Web3D worlds through automatically generated tours. In *Proceedings of ACM Conference on 3D Web Technology 2003*, pages 27–38, 2003.

[31] Luca Chittaro and Ivan Scagnetto. Is semitransparency useful for navigating virtual environments? In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology 2001*, pages 159–166, 2001.

[32] Mei C. Chuah, Steven F. Roth, Joe Mattis, and John Kolojejchick. SDM: Selective dynamic manipulation of visualizations. In *Proceedings of the*

ACM Symposium on User Interface Software and Technology 1995, pages 61–70, 1995.

[33] Chris Coffin and Tobias Höllerer. Interactive perspective cut-away views for general 3D scenes. In *Proceedings of the IEEE Symposium on 3D User Interfaces 2006*, pages 25–28, 2006.

[34] Thomas H. Cormen, Charles E. Lesierson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms.* MIT Press, Cambridge, MA, second edition, 2001.

[35] Carolina Cruz-Neira, Daniel J. Sandin, Thomas A. DeFanti, Robert V. Kenyon, and John C. Hart. The CAVE: audio visual experience automatic virtual environment. *Communications of the ACM*, 35(6):64–72, June 1992.

[36] Rudolph P. Darken and Barry Peterson. Spatial orientation, wayfinding, and representation. In Kay M. Stanney, editor, *Handbook of Virtual Environment Technology.* Lawrence Erlbaum Associates, 2001.

[37] Rudolph P. Darken and John L. Sibert. Wayfinding strategies and behaviors in large virtual worlds. In *Proceedings of the ACM CHI'96 Conference on Human Factors in Computing Systems*, pages 142–149, 1996.

[38] Maylis Delest, Tamara Munzner, David Auber, and Jean-Philippe Domenger. Exploring InfoVis publication history with Tulip. InfoVis 2004 Contest.

[39] Peter J. Denning. The ACM digital library goes live. *Communications of the ACM*, 40(7):28–29, 1997.

[40] Giuseppe DiBattista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs.* Prentice Hall, 1999.

[41] Paul J. Diefenbach. *Pipeline Rendering: Interaction and Realism through Hardware-Based Multi-Pass Rendering.* Ph.D. thesis, Computer Graphics, University of Pennsylvania, 1996.

[42] Joachim Diepstraten, Daniel Weiskopf, and Thomas Ertl. Transparency in interactive technical illustrations. *Computer Graphics Forum*, 21(3):317–325, 2002.

[43] Joachim Diepstraten, Daniel Weiskopf, and Thomas Ertl. Interactive cut-away rendering. In *Proceedings of Eurographics 2003*, pages 523–532, 2003.

[44] Niklas Elmqvist. BalloonProbe: Reducing occlusion in 3D using interactive space distortion. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology 2005*, pages 134–137, 2005.

[45] Niklas Elmqvist and Philippas Tsigas. Causality visualization using animated growing polygons. In *Proceedings of the IEEE Symposium on Information Visualization 2003*, pages 189–196, 2003.

[46] Niklas Elmqvist and Philippas Tsigas. Growing squares: Animated visualization of causal relations. In *Proceedings of the ACM Symposium on Software Visualization 2003*, pages 17–26, 2003.

[47] Niklas Elmqvist and Philippas Tsigas. Animated visualization of causal relations through growing 2D geometry. *Information Visualization*, 3(3):154–172, 2004.

[48] Niklas Elmqvist and Philippas Tsigas. Citewiz: A tool for the visualization of scientific citation networks. Technical Report 2004-05, Chalmers University of Technology, 2004.

[49] Niklas Elmqvist and Philippas Tsigas. On navigation guidance in 3D environments. Technical Report 2006-19, Chalmers University of Technology, 2006.

[50] Niklas Elmqvist and Philippas Tsigas. View projection animation for occlusion reduction. In *Proceedings of the ACM Conference on Advanced Visual Interfaces 2006*, pages 471–475, 2006.

[51] Niklas Elmqvist and Philippas Tsigas. A taxonomy of 3D occlusion management techniques. In *Proceedings of the IEEE Conference on Virtual Reality 2007*, to appear.

[52] Niklas Elmqvist and Mihail Eduard Tudoreanu. Evaluating the effectiveness of occlusion reduction techniques for 3D virtual environments. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology 2006*, pages 9–18, 2006.

[53] T. Todd Elvins, David R. Nadeau, and David Kirsh. Worldlets – 3D thumbnails for wayfinding in virtual environments. In *Proceedings of the ACM Symposium on User Interface Software and Technology 1997*, pages 21–30, 1997.

[54] Cass Everitt. Interactive order-independent transparency. NVIDIA Corporation, 2001. See *http://developer.nvidia.com*.

[55] Jean-Daniel Fekete, Georges Grinstein, and Catherine Plaisant. InfoVis 2004 Contest: The History of InfoVis, 2004. *http://www.cs.umd.edu/hcil/iv04contest/*.

[56] James D. Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes. *Computer Graphics: Principles and Practice.* Addison-Wesley, Reading MA, 1990.

[57] Bert Freudenberg, Maic Masuch, and Thomas Strothotte. Real-time halftoning: A primitive for non-photorealistic shading. In *Proceedings of the Eurographics Workshop on Rendering 2002*, pages 227–232, 2002.

[58] Shinji Fukatsu, Yoshifumi Kitamura, Toshihiro Masaki, and Fumio Kishino. Intuitive control of "bird's eye" overview images for navigation in an enormous virtual environment. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology 1998*, pages 67–76, 1998.

[59] George W. Furnas. Generalized fisheye views. In *Proceedings of the ACM CHI'86 Conference on Human Factors in Computer Systems*, pages 16–23, 1986.

[60] Paul Fussell. *The Norton Book of Travel*, chapter The Eighteenth Century and the Grand Tour. W.W. Norton and Co., 1987.

[61] Tinsley A. Galyean. Guided navigation of virtual environments. In *Proceedings of the ACM Symposium on Interactive 3D Graphics 1995*, pages 103–104, 1995.

[62] James J. Gibson. *The Perception of the Visual World.* Houghton Mifflin, 1950.

[63] C. Lee Giles, Kurt Bollacker, and Steve Lawrence. CiteSeer: An automatic citation indexing system. In *Proceedings of the ACM Conference on Digital Libraries 1998*, pages 89–98, 1998.

[64] Michael Gleicher and Andrew Witkin. Through-the-lens camera control. In *Computer Graphics (SIGGRAPH '92 Proceedings)*, pages 331–340, 1992.

[65] E. Bruce Goldstein. *Sensation and Perception.* Wadsworth-Thomson, 6th edition, 2002.

[66] Amy Gooch, Bruce Gooch, Peter Shirley, and Elaine Cohen. A non-photorealistic lighting model for automatic technical illustration. In *Proceedings of the ACM Conference on Computer Graphics (SIGGRAPH '98)*, pages 447–452, 1998.

[67] Bruce Gooch, Peter-Pike J. Sloan, Amy Gooch, Peter Shirley, and Richard F. Riesenfeld. Interactive technical illustration. In *Proceedings of the ACM Symposium on Interactive 3D 1999*, pages 31–38, 1999.

[68] Tovi Grossman, Ravin Balakrishnan, Gordon Kurtenbach, George W. Fitzmaurice, Azam Khan, and William Buxton. Interaction techniques for 3D modeling on large displays. In *Proceedings of the ACM Symposium on Interactive 3D Graphics 2001*, pages 17–23, 2001.

[69] Tovi Grossman, Ravin Balakrishnan, Gordon Kurtenbach, George W. Fitzmaurice, Azam Khan, and William Buxton. Creating principal 3D curves with digital tape drawing. In *Proceedings of the ACM CHI 2002 Conference on Human Factors in Computing Systems*, pages 121–128, 2002.

[70] Carl Gutwin, Jeff Dyck, and Chris Fedak. The effects of dynamic transparency on targeting performance. In *Proceedings of Graphics Interface 2003*, pages 105–112, 2003.

[71] Andrew J. Hanson and Eric A. Wernert. Constrained 3D navigation with 2D controllers. In *Proceedings of the IEEE Conference on Visualization 1997*, pages 176–182, 1997.

[72] Beverly L. Harrison, Gordon Kurtenbach, and Kim J. Vicente. An experimental evaluation of transparent user interface tools and information content. In *Proceedings of the ACM Symposium on User Interface Software and Technology 1995*, pages 81–90, 1995.

[73] Michael T. Heath. Visual animation of parallel algorithms for matrix computations. In *Proceedings of the Fifth Distributed Memory Computing Conference*, pages 1213–1222, April 1990.

[74] Michael T. Heath and Jennifer A. Etheridge. Visualizing the performance of parallel programs. *IEEE Software*, 8(5):29–39, September 1991.

[75] Matthias Hemmje, Clemens Kunkel, and Alexander Willett. LyberWorld – A visualization user interface supporting fulltext retrieval. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval 1994*, pages 249–259, 1994.

[76] Aaron Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. In *Proceedings of the ACM Conference on Computer Graphics (SIGGRAPH '98)*, pages 453–460, 1998.

[77] Takeo Igarashi, Rieko Kadobayashi, Kenji Mase, and Hidehiko Tanaka. Path drawing for 3D walkthrough. In *Proceedings of the ACM Symposium on User Interface Software and Technology 1998*, pages 173–174, 1998.

[78] Tobias Isenberg, Bert Freudenberg, Nick Halper, Stefan Schlechtweg, and Thomas Strothotte. A developer's guide to silhouette algorithms for polygonal models. *IEEE Computer Graphics and Applications*, 23(4):28–37, 2003.

[79] Edward W. Ishak and Steven K. Feiner. Interacting with hidden content using content-aware free-space transparency. In *Proceedings of the ACM Symposium on User Interface Software and Technology 2004*, pages 189–192, 2004.

[80] William H. Ittelson. *Visual Space Perception*. Springer Publishing Company, 1960.

[81] Arie Kaufman and Eyal Shimony. 3D scan-conversion algorithms for voxel-based graphics. In *Proceedings of the ACM Workshop on Interactive 3D Graphics 1986*, pages 45–75, 1986.

[82] Douglas S. Kay and Donald P. Greenberg. Transparency for computer synthesized images. In *Computer Graphics (SIGGRAPH '79 Proceedings)*, pages 158–164, 1979.

[83] Weimao Ke, Katy Borner, and Lalitha Viswanath. Major information visualization authors, papers and topics in the ACM library. InfoVis 2004 Contest.

[84] Daniel A. Keim, Helmut Barro, Christian Panse, Jorn Scheidewind, and Mike Sips. Exploring and visualizing the history of InfoVis. InfoVis 2004 Contest.

[85] Michael M. Kessler. Bibliographic coupling between scientific papers. *American Documentation*, 14(1):10–25, January 1963.

[86] Boris Koldehofe, Marina Papatriantafilou, and Philippas Tsigas. Distributed algorithms visualisation for educational purposes. In *Proceedings of the SIGCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education 1999*, pages 103–106, 1999.

[87] Eileen Kraemer and John T. Stasko. Creating an accurate portrayal of concurrent executions. *IEEE Concurrency*, 6(1):36–46, January/March 1998.

[88] John Lamping and Ramana Rao. The Hyperbolic Browser: A focus + context technique for visualizing large hierarchies. *Journal of Visual Languages and Computing*, 7(1):33–35, 1996.

[89] Leslie Lamport. Time, clocks and the ordering of events in distributed systems. *Communications of the ACM*, 21(7):558–564, 1978.

[90] Haim Levkowitz and Gabor T. Herman. Color scales for image data. *IEEE Computer Graphics and Applications*, 12(1):72–80, January 1992.

[91] Haim Levkowitz, Richard A. Holub, Gary W. Meyer, and Philip K. Robertson. Color versus black and white in visualization. *IEEE Computer Graphics and Applications*, 12(4):20–22, July 1992.

[92] Wilmot Li, Maneesh Agrawala, and David Salesin. Interactive image-based exploded view diagrams. In *Proceedings of Graphics Interface 2004*, pages 203–212, 2004.

[93] Yong Kui Liu, Borut Zalik, and Hongji Yang. An integer one-pass algorithm for voxel traversal. *Computer Graphics Forum*, 23(2):167–172, 2004.

[94] Julian Looser, Mark Billinghurst, and Andy Cockburn. Through the looking glass: the use of lenses as an interface tool for augmented reality interfaces. In *Proceedings of GRAPHITE 2004*, pages 204–211, 2004.

[95] Peter Lyman and Hal R. Varian. How much information?, 2000. http://www2.sims.berkeley.edu/research/projects/how-much-info/.

[96] Peter Lyman and Hal R. Varian. How much information 2003?, 2003. http://www2.sims.berkeley.edu/research/projects/how-much-info-2003/.

[97] Jock Mackinlay. Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics*, 5(2):110–141, 1986.

[98] Jock D. Mackinlay, Stuart K. Card, and George G. Robertson. Rapid controlled movement through a virtual 3D workspace. In *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 171–176, August 1990.

[99] Jock D. Mackinlay, Ramana Rao, and Stuart K. Card. An organic user interface for searching citation links. In *Proceedings of the ACM CHI'95 Conference on Human Factors in Computing Systems*, pages 67–73, 1995.

[100] Jock D. Mackinlay, George G. Robertson, and Stuart K. Card. The Perspective Wall: Detail and context smoothly integrated. In *Proceedings of the ACM CHI'91 Conference on Human Factors in Computing Systems*, pages 173–179, 1991.

[101] Abraham Mammen. Transparency and antialiasing algorithms implemented with the virtual pixel maps technique. *IEEE Computer Graphics and Applications*, 9(4):43–55, July 1989.

[102] Michael J. McGuffin, Liviu Tancau, and Ravin Balakrishnan. Using deformations for browsing volumetric data. In *Proceedings of the IEEE Conference on Visualization 2003*, pages 401–408, 2003.

[103] James C. Michener and Ingrid B. Carlbom. Natural and efficient viewing parameters. In *Computer Graphics (SIGGRAPH '80 Proceedings)*, pages 238–245, July 1980.

[104] David Modjeska, Vassilios Tzerpos, Petros Faloutsos, and Michalis Faloutsos. BIVTECI: A bibliographic visualization tool. In *Proceedings of the 1996 Conference of the Centre of Advanced Studies on Collaborative Research*, page 28, 1996.

[105] Yoram Moses, Zvi Polunsky, and Ayellet Tal. Algorithm visualization for distributed environments. In *Proceedings of the IEEE Symposium on Information Visualization 1998*, pages 71–78, 1998.

[106] Marc Nienhaus and Jürgen Döllner. Blueprints: Illustrating architecture and technical parts using hardware-accelerated non-photorealistic rendering. In *Proceedings of Graphics Interface 2004*, pages 49–56, 2004.

[107] Donald A. Norman. *The Psychology of Everyday Things*. Basic Books, New York, 1988.

[108] Donald A. Norman. Cognition in the head and in the world. *Cognitive Science*, 17(1):1–6, January-March 1993.

[109] Takanori Okoshi. *Three-dimensional Imaging Techniques*. Academic Press, 1976.

[110] Ken Perlin and David Fox. Pad: An alternative approach to the computer interface. In *Proceedings of Computer Graphics (SIGGRAPH 93)*, pages 57–64, 1993.

[111] Jeffrey S. Pierce, Matthew Conway, Maarten van Dantzich, and George Robertson. Tool spaces and glances: Storing, accessing, and retrieving objects in 3D desktop applications. In *Proceedings of the ACM Symposium on Interactive 3D Graphics 1999*, pages 163–168, 1999.

[112] William Playfair. *The Commercial and Political Atlas*. London, 1786.

[113] Nicholas F. Polys and Doug A. Bowman. Design and display of enhancing information in desktop information-rich virtual environments: challenges and techniques. *Virtual Reality*, 8(1):41–54, 2004.

[114] Zachary Pousman and John Stasko. A taxonomy of ambient information systems: Four patterns of design. In *Proceedings of the ACM Conference on Advanced Visual Interfaces 2006*, pages 67–74, 2006.

[115] Emil Praun, Hugues Hoppe, Matthew Webb, and Adam Finkelstein. Real-time hatching. In *Proceedings of the ACM Conference on Computer Graphics (SIGGRAPH 2001)*, pages 581–581, 2001.

[116] Gonzalo Ramos, George Robertson, Mary Czerwinski, Desney Tan, Patrick Baudisch, Ken Hinckley, and Maneesh Agrawala. Tumble! Splat! Helping users access and manipulate occluded content in 2D drawings. In *Proceedings of the ACM Conference on Advanced Visual Interfaces 2006*, pages 428–435, 2006.

[117] Ramana Rao and Stuart K. Card. The Table Lens: Merging graphical and symbolic representations in an interactive focus+context visualization for tabular information. In *Proceedings of the ACM CHI'94 Conference on Human Factors in Computing Systems*, pages 318–322, 1994.

[118] Thomson ResearchSoft. RefViz, 2006. *http://www.refviz.com*.

[119] George G. Robertson, Stuart K. Card, and Jock D. Mackinlay. Information visualization using 3D interactive animation. *Communications of the ACM*, 36(4):56–71, April 1993.

[120] George G. Robertson, Jock D. Mackinlay, and Stuart K. Card. Cone trees: Animated 3D visualizations of hierarchical information. In *Proceedings of the ACM CHI'91 Conference on Human Factors in Computing Systems*, pages 189–194, 1991.

[121] Roy A. Ruddle. The effect of trails on first-time and subsequent navigation in a virtual environment. In *Proceedings of the IEEE Conference on Virtual Reality 2005*, pages 115–122, 2005.

[122] Mike Scaife and Yvonne Rogers. External cognition: How do graphical representations work? *International Journal of Human-Computer Studies*, 45(2):185–213, 1996.

[123] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the IEEE Symposium on Visual Languages*, pages 336–343, September 3–6 1996.

[124] Karan Singh. A fresh perspective. In *Proceedings of Graphics Interface 2002*, pages 17–24, 2002.

[125] Karan Singh and Ravin Balakrishnan. Visualizing 3D scenes using non-linear projections and data mining of previous camera movements. In *Proceedings of AFRIGRAPH 2004*, pages 41–48, 2004.

[126] Karan Singh, Cindy Grimm, and Nisha Sudarsanam. The IBar: a perspective-based camera widget. In *Proceedings of the ACM Symposium on User Interface Software and Technology 2004*, pages 95–98, 2004.

[127] Henry G. Small. Co-citation in the scientific literature: A new measure of the relationship between two documents. *Journal of the American Society for Information Science*, 24(4):265–269, July-August 1973.

[128] David Socha, Mary L. Bailey, and David Notkin. Voyeur: Graphical views of parallel programs. In *Proceedings of the ACM SIGPLAN/SIGOPS Workshop on Parallel and Distributed Debugging 1989*, pages 206–215, 1989.

[129] Deyang Song and Michael Norman. Nonlinear interactive motion control techniques for virtual space navigation. In *Proceedings of IEEE Virtual Reality Annual International Symposium 1993*, pages 111–117, 1993.

[130] Henry Sonnet, M. Sheelagh T. Carpendale, and Thomas Strothotte. Integrating expanding annotations with a 3D explosion probe. In *Proceedings of the ACM Conference on Advanced Visual Interfaces 2004*, pages 63–70, 2004.

[131] John T. Stasko and Eileen Kraemer. A methodology for building application-specific visualizations of parallel programs. *Journal of Parallel and Distributed Computing*, 18(2):258–264, June 1993.

[132] Richard Stoakley, Matthew J. Conway, and Randy Pausch. Virtual Reality on a WIM: Interactive worlds in miniature. In *Proceedings of the ACM CHI'95 Conference on Human Factors in Computing Systems*, pages 265–272, 1995.

[133] Ivan E. Sutherland. A head-mounted three dimensional display. In *Proceedings of the AFIPS Conference 1968*, pages 757–764, 1968.

[134] Desney S. Tan, George G. Robertson, and Mary Czerwinski. Exploring 3D navigation: Combining speed-coupled flying with orbiting. In *Proceedings of the ACM CHI 2001 Conference on Human Factors in Computing Systems*, pages 418–425, 2001.

[135] Edward C. Tolman. Cognitive maps in rats and men. *The Psychological Review*, 55(4):189–208, July 1948.

[136] Brad Topol, John T. Stasko, and Vaidy Sunderam. PVaniM: a tool for visualization in network computing environments. *Concurrency: Practice and Experience*, 10(14):1197–1222, 1998.

[137] Edward R. Tufte. *The Visual Display of Quantitative Information.* Graphics Press, Cheshire, CT, 1983.

[138] John Viega, Matthew J. Conway, George Williams, and Randy Pausch. 3D magic lenses. In *Proceedings of the ACM Symposium on User Interface Software and Technology 1996*, pages 51–58, 1996.

[139] Norman G. Vinson. Design guidelines for landmarks to support navigation in virtual environments. In *Proceedings of the ACM CHI'99 Conference on Human Factors in Computing Systems*, pages 278–285, 1999.

[140] Ivan Viola, Armin Kanitsar, and Eduard Gröller. Importance-driven volume rendering. In *Proceedings of the IEEE Conference on Visualization 2004*, pages 139–145, 2004.

[141] Colin Ware and Danny R. Jessome. Using the bat: A six-dimensional mouse for object placement. *IEEE Computer Graphics and Applications*, 8(6):65–70, November 1988.

[142] Colin Ware, Eric Neufeld, and Lyn Bartram. Visualizing causal relations. In *Proceedings of the IEEE Symposium on Information Visualization 1999 (Late Breaking Hot Topics)*, pages 39–42, 1999.

[143] Colin Ware and Steven Osborne. Exploration and virtual camera control in virtual three dimensional environments. *Computer Graphics*, 24(2):175–183, 1990.

[144] Eric A. Wernert and Andrew J. Hanson. A framework for assisted exploration with collaboration. In *Proceedings of the IEEE Conference on Visualization 1999*, pages 241–248, 1999.

[145] Nelson Wong, M. Sheelagh T. Carpendale, and Saul Greenberg. EdgeLens: An interactive method for managing edge congestion in graphs. In *Proceedings of the IEEE Symposium on Information Visualization 2003*, pages 51–58, 2003.

[146] Günther Wyszecki and W. S. Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulae.* John Wiley & Sons, second edition, 1991.

[147] Jing Yang, Matthew O. Ward, and Elke A. Rundensteiner. InterRing: An interactive tool for visually navigating and manipulating hierarchical structures. In *Proceedings of the IEEE Symposium on Information Visualization 2002*, pages 77–84, 2002.

Part III

Appendices

Appendix A

View Projection Animation User Study

A.1 Script

0. Preparations

   0.1. Post-test questionnaire (1 copy)

1. Introduction (5 minutes)

   1.1. Experimenter welcomes subject

   1.2. Experimenter gives brief explanation of study

      a) Primary purpose: Evaluation of the impact of inter-object occlusion on object discovery efficiency in 3D environments
      b) Secondary purpose: Comparative evaluation of normal perspective view as opposed to a projection morphing view for occlusion reduction (improving object discovery efficiency)
      c) Occlusion is not only a geometrical measure, but also a subjective measure based on perception and recognition of partially occluded objects
      d) Conventional 3D perspective view mode versus a new interactive projection morph mode
      e) Each subject will use both of the two view modes to solve a set of tasks (between-subjects)
      f) Each set of tasks represents a combination of view mode, camera mode and dataset size and includes 20 similar tasks with randomized data
      g) Hardware: Pentium III desktop computer with 3D accelerator

   h) Software: information visualization application (PMorph) running on the Linux operating system
   i) Study data will be reported anonymously in a PhD thesis and possibly in an academic paper

1.3. Experimenter gives brief background on object discovery and occlusion in 3D environments

   a) Visualization applications represent information through **marks** with various graphical properties dispersed throughout a visual substrate
   b) For 3D visualizations, the visual substrate is a subset of the full 3D set, and marks are 3D objects located in this subspace
   c) The nature of the human visual system means that 3D objects can easily occlude (hide) other 3D objects in the same environment, partially or totally
   d) Total occlusion hampers object discovery for the user, i.e. the ability to find a specific object in an environment

2. Training (5 minutes)

2.1. Experimenter demonstrates the application the subject will use

2.2. Object search task in a given 3D environment

   a) Use the "sandbox" test environment
   b) Orbiting the camera (note that camera may be fixed)
   c) Object search tasks (red, green and blue objects)
   d) Projection morphing and its uses

2.3. Subject is allowed up to 5 minutes of practice using a test environment

   a) Use the "sandbox" test environment
   b) Subject decides when ready to proceed

3. Task explanation (5 minutes)

3.1. Subject will use both view modes

3.2. Subject is to perform a simple object search task in a given 3D environment (camera and density parameters vary)

   a) Application will prompt the user to search for objects of a given color (red, green or blue) in a set of unimportant objects (these are always adjusted to not have colors too similar to the one being searched for)
   b) Use the view window to count the number of objects of the given color

    c) Depending on the task set, the user may be able to move (orbit) the camera and morph the projection to help solve the task
    d) Enter the number of objects in the test prompt window and move on to the next task

  3.3. Subtasks and correctness is automatically timed by the application

    a) The idea is to work fast without being sloppy
    b) Better to be correct than to be fast

  3.4. Subject will respond to a short questionnaire at the end of the test

  3.5. Questions to the experimenter are acceptable at any time

4. Task execution (5-10 minutes per task set) — [repeated for all eight conditions]

    4.1. Experimenter decides which task set to run (randomize order for each user, use increasing object density)
    4.2. Experimenter initializes a task set of the right view, camera and density parameters
    4.3. Subject is allowed to proceed with solving the tasks
    4.4. Experimenter does not intervene during each task set, only answers questions and resolves any technical issues

5. Post-test questionnaire

    5.1. Experimenter gives the post-test questionnaire form to the subject
    5.2. Subject fills out the questionnaire

6. Conclusion

    6.1. Experimenter thanks subject for participation in the study
    6.2. Experimenter issues the agreed compensation to the subject (if any)
    6.3. Experimenter asks subject if he or she has any questions and answers them accordingly
    6.4. Subject departs

A.2 Scenarios

A.2.1 Introduction

Basic introduction of the scenario given to the subject by the experimenter:

> You will be shown a simple 3D environment filled with boxes in different colors. These boxes have been randomly placed in the environment and been given a random size and rotation. A few of the target boxes are *targets* and have a specific color, the rest are *distractors*. Your task is to **count** the target objects in the environment.
>
> The target color changes for each task, but is either bright red, bright blue, or bright green. For each task, you will be shown a new environment with new boxes, and the application will tell you in a dialog box which color you are looking for.

A.2.2 Task Set Instructions

Instructions for a whole task set given to the subject by the application:

> This test will evaluate the impact of occlusion on object discovery. Your task is to count the number of objects of a given color (red, green, blue) and enter the value in the field below. You will be given 10 such tasks. Press Enter in the text field or the "next" button to proceed when you are ready.
>
> PMorph: m
> Camera: c

- m – view projection animation method ["on", "off"]
- c – camera type ["fixed", "dynamic"]

A.2.3 Task Instructions

Instructions for a single task given to the subject by the application:

> [#n] Count the number of c objects in this scene.

- n – task number [$1 \ldots 10$]
- c – target color ["red", "green", "blue"]

A.3 Post-Test Questionnaire

1. Which modes did you like more with respect to **ease of use**?

   1.1. Projection modes:
   ☐ Normal perspective mode
   ☐ View projection animation mode
   ☐ Undecided

1.2. Camera modes:
☐ Fixed camera
☐ Movable camera
☐ Undecided

2. Which modes did you like more with respect to **efficiency** of solving the tasks?

2.1. Projection modes:
☐ Normal perspective mode
☐ View projection animation mode
☐ Undecided

2.2. Camera modes:
☐ Fixed camera
☐ Movable camera
☐ Undecided

3. Which modes did you like more with respect to **enjoyment**?

3.1. Projection modes:
☐ Normal perspective mode
☐ View projection animation mode
☐ Undecided

3.2. Camera modes:
☐ Fixed camera
☐ Movable camera
☐ Undecided

4. Which mode helped you feel the most **confident** about having discovered all objects in the scene?
☐ Normal perspective – fixed camera
☐ Normal perspective – movable camera
☐ View projection animation – fixed camera
☐ View projection animation – movable camera
☐ Undecided

5. Which mode did you feel was the **fastest** to use?
☐ Normal perspective – fixed camera
☐ Normal perspective – movable camera
☐ View projection animation – fixed camera
☐ View projection animation – movable camera
☐ Undecided

6. Overall, which mode would you choose for performing this task in your daily work?
   ☐ Normal perspective – fixed camera
   ☐ Normal perspective – movable camera
   ☐ View projection animation – fixed camera
   ☐ View projection animation – movable camera
   ☐ Undecided

Appendix B

BalloonProbe User Study[1]

B.1 Script

0. Preparations

   0.1. Consent form (2 copies)

   0.2. Post-test questionnaire (1 copy)

   0.3. Answer sheets (5 copies)

   0.4. Experimenter decides which task group the subject will belong to (round-robin assignment)

      a) Task groups: count, pattern, relate

   0.5. Experimenter prepares the application with the correct task

1. Introduction

   1.1. Experimenter welcomes subject

   1.2. Experimenter gives the subject the consent form

      a) Experimenter asks the subject to read the consent form
      b) Experimenter asks that the subject should sign the consent form if, and only if, he or she agrees with its contents
      c) Experimenter gives the subject a copy of the consent form for future reference

   1.3. Experimenter gives brief explanation of study

[1]Note that this user study was originally conducted at the University of Arkansas at Little Rock by Mihail Eduard Tudoreanu. The script found here is the author's own version, created for a complementary study using the same design and software platform.

a) Purpose: “Our research deals with developing techniques to help users understand and interact with 3D environments. This is generally a tough problem due to the scenery and the objects themselves getting in the way of each other. The purpose of this study is to compare the performance of five different techniques for improving 3D interaction despite these problems.”
b) Applications: “Many practical 3D worlds, such as virtual environments, 3D simulations, 3D visualizations, modeling, and even 3D games have this problem.”
c) Techniques: “Four of the techniques we will be testing are designed to help a user discover and access objects in the environment. The first two are called BalloonProbes, the next two are 3D fisheyes. The final one is a base case, simple 3D flying navigation with no special support for discovery and access.”
d) Research question: “Basically, what we want to find out with this study is whether different techniques are good at different types of tasks. This could help us develop a set of design guidelines for when to use a specific technique. We also want to see whether actually having an interaction technique like this makes any sense; therefore, we have included the base case into the study as well.”
e) Procedure: “We will be testing the five different techniques in two types of worlds and for three different tasks. Each participant will be using all five techniques in both worlds. However, you will only be performing one of the tasks. The task group you have been assigned to has been decided randomly by the experimenter (me).”
f) Tasks: “Each combination of technique and world is performed four different times (i.e. four trials). The task you will be doing is done as follows:”
   - Count: “You will be shown the world with a number of objects scattered randomly around the environment. The objects are simple 3D primitives such as spheres, boxes, and cones, and have a random size, color, and rotation. Most objects are distractors and not important for the task, but some are targets. Your task is to count the total number of targets that you can find in the environment. The targets you are looking for are yellow cones.”
   - Pattern: “You will be shown the world with a number of objects scattered randomly around the environment. The objects are simple 3D primitives such as spheres, boxes, and cones, and have a random size, color, and rotation. Most objects are distractors and not important for the task, but some are tar-

gets. Your task is to identify which pattern the targets form. The pattern is always one of the capital letters C, K, R, X, or Y. The targets you are looking for are yellow cones."

- Relate: "You will be shown the world with a number of objects scattered randomly around the environment. The objects are simple 3D primitives such as spheres, boxes, and cones, and have a random size, color, and rotation. Most objects are distractors and not important for the task, but some are targets. Yellow cones and green boxes are targets; they are found in pairs around the environment. A third unknown target is 'spying' on the cone and box—you can find it in close proximity of the pairs around the environment. Your task is to identify the spy (color and type)."

g) Worlds: "The two worlds you will be exploring in this study is either a very simple abstract 3D environment filled with 3D primitives such as cones, boxes, and spheres, or a one-floor 3D building consisting of rooms, walls, and doors."

h) Measurements: "The test application will be measuring the time for each individual trial for each environment. Furthermore, we will also record your accuracy in finding the correct answer for each trial. Therefore, it is important to work both efficiently and thoroughly."

i) Ordering: "The order you receive the scenarios and the techniques will be randomly determined by the experimenter (me) to avoid effects of practice."

j) Training: "Before starting the experiment itself, you will be given a short training session where you are allowed to familiarize yourself with the two different environments and the tasks that you will be asked to perform in each of them. You may ask questions at any point during this training session. During the test itself, I will only answer questions relating to technical or study issues."

k) Experimenter asks whether the subject has any questions and answers them accordingly

1.4. Experimenter determines ordering of worlds and techniques (randomly or by design)

a) Worlds: abstract, architectural

b) Techniques: none, sphere, wedge, scale, transparency

2. Training

2.1. Experimenter explains the basic idea behind each technique

a) Navigation: "Basic navigation is done by controlling the viewpoint as if it was attached to a helicopter flying through the 3D world."

b) BalloonProbe: "With the BalloonProbe technique, you have access to a portable force field that is connected to your wand. You can inflate and deflate it at will. By inflating the force field, you will push away all of the objects that fall inside the field. This allows you to see in areas where there is a high concentration of objects. You can change the size of the probe using the '1' and '2' keys."

- Sphere: "Spherical BalloonProbe uses a standard sphere-shaped balloon."
- Wedge: "Wedge BalloonProbe uses a wedge-shaped balloon. The effect is a little like parting branches."

c) Fisheye: "3D fisheye views are designed to distort the space in a way so that nearby objects are shown with full detail and more distant ones are shown with less detail. This makes it possible to get an overview of a large dataset while still retaining a detailed view of the current focus. In our version of the fisheye, the wand you hold in your hand is the focus."

- Scale-based: "The scale-based fisheye controls detail using the size of the objects. Nearby objects are shown at full size, whereas more distant objects become smaller and smaller."
- Transparency-based: "The transparency-based fisheye controls detail using the transparency of the objects. Nearby objects are shown as normal, whereas more distant objects become more and more transparent so that you can see through them."

2.2. Experimenter starts the training application

2.3. Experimenter demonstrates the operation of the different techniques (including navigation) and lets the subject try himself or herself

a) Navigation

- Move around the CAVE physically to navigate locally
- Use the navigation controls on the wand to navigate globally
- Move the wand to change the position of the 3D cursor (the focus point for the techniques)
- Remember that you can both sit or stand when performing the navigation—if you prefer to sit, you can use a chair

b) BalloonProbe

- Move the wand to change the center point of the BalloonProbe

- Rotate the wand to change the orientation of the Balloon-Probe (only relevant for the wedge probe)
- For this test, the probe is always inflated
- Change the size of the probe using the '1' and '2' buttons

c) 3D fisheye

- Move the wand to change the focus of the fisheye
- Change the intensity of the technique (the impact of distance from focus on detail level) using the '1' and '2' buttons

d) Task procedure

i) Each trial starts with a blank screen giving instructions on world and technique

ii) The trial is started by hitting a button on the wand (this starts the timer)

iii) The trial can be paused and resumed (including the timer) at any time

iv) The trial is ended by hitting a button on the wand (this stops the timer)

e) Experimenter asks whether the subject has any questions and answers them accordingly

3. Scenario execution — [repeated for all five techniques]

3.1. Experimenter informs subject of current technique to use and reminds the subject of its operation

3.2. World execution – [repeated for both worlds]

a) Experimenter launches the application for the current world and technique

b) Trial execution – [repeated for four trials]

i) A new trial is generated by the application

ii) Subject performs the task in the given world and technique

iii) Subject marks answers on the answer sheet for the corresponding technique when he or she is done

4. Post-test questionnaire

4.1. Experimenter gives the post-test questionnaire form to the subject

4.2. Subject fills out the questionnaire

5. Conclusion

5.1. Experimenter thanks subject for participation in the study

   5.2. Experimenter issues the agreed compensation to the subject (if any)
   5.3. Experimenter asks subject if he or she has any questions and answers them accordingly
   5.4. Subject departs

6. Epilogue
   6.1. Experimenter collects post-test questionnaire
   6.2. Experimenter copies the log files from the specific directories to the results directory

B.2 Scenarios

Scenarios are given in the text of the script in Section B.1 above.

B.3 Post-Test Questionnaire

1. Did you spend most of the time sitting or standing? Why?
2. Did you modify the size/transparency of various techniques (by pressing buttons '1' or '2')? Did you end up making a lot of changes to the size/transparency?
3. Please write any other comments here.

Appendix C

Dynamic Transparency User Study

C.1 Script

0. Preparations
    - 0.1. Consent form (2 copies)
    - 0.2. Post-test questionnaire (1 copy)
    - 0.3. Empty log files for both applications (`stats.dat`)
1. Introduction (5 minutes)
    - 1.1. Experimenter welcomes subject
    - 1.2. Experimenter gives the subject the consent form
    - 1.3. Experimenter asks the subject to read the consent form
    - 1.4. Experimenter asks that the subject should sign the consent form if, and only if, he or she agrees with its contents
    - 1.5. Experimenter gives the subject a copy of the consent form for future reference
    - 1.6. Experimenter gives brief explanation of study
        - a) Purpose: "The purpose of this study is to compare a new visualization technique against traditional 3D navigation techniques. Our new method is called dynamic transparency, and gives the user X-ray vision in the computer world, allowing you to see through walls and objects."
        - b) Methods: "We will be testing this new technique in two different types of environments, one based on a large collection of simple 3D objects, the other based on a building-like environment of rooms,

    doors and hallways. You will be solving ten simple tasks for each of the two types of environments. You will do this both using X-ray vision, as well as without X-ray vision."

    c) Task: "The task you will be solving is a simple search task. In both cases, you are looking for a specific object."
    d) Scenario 1 (abstract): "In the first scenario, you will encounter a large number of simple objects in many colors. Your first task here is to find and count all of the red objects. Your second task here is to detect the pattern the red cones form (this pattern is always a capital letter, either C, K, R, X or Y)."
    e) Scenario 2 (walkthrough): "In the second scenario, you will be exploring a building environment using the mouse and keyboard, looking for one specific object given in the upper left corner of the window. In a second task, you will have to count the number of occurrences of the specified object."
    f) Ordering: "The order you receive the two scenarios as well as the order in which you use X-ray vision or not will be randomly determined by the experimenter (me)."
    g) Training: "Before starting the experiment itself, you will be given a short training session where you are allowed to familiarize yourself with the two different environments and the tasks that you will be asked to perform in each of them. You may ask questions at any point during this training session. During the test itself, I will only answer questions relating to technical or study issues."
    h) Data collection: "The test applications will record your performance, both in terms of the time taken to complete each task as well as how close your answer is to the correct answer. All data will be kept strictly confidential and will be identified with a code number and not your name.
    i) Experimenter asks whether the subject has any questions and answers them accordingly

  1.7. Experimenter determines ordering of conditions (randomly or by design)

    a) Abstract (scenario 1) versus walkthrough (scenario 2) environment
    b) Use X-ray or do not use X-ray (within each scenario)

2. Scenario execution (25 minutes per scenario) — [repeated for both scenarios]

  2.1. Experimenter informs subject of current scenario (abstract or walkthrough)

2.2. Training (5 minutes)

   a) Experimenter starts the trial application for the current scenario
      - Scenario 1: `Game1ReleaseA/trial.bat`
      - Scenario 2: `Game2ReleaseA/trial.bat`

   b) Experimenter demonstrates the interface of the scenario application
      i) Scenario 1 (abstract 3D environment)
         - Change the viewpoint by pressing the left mouse button and moving the mouse
         - Indicate you are done by pressing the Space bar

      ii) Scenario 2 (walkthrough environment)
         - Move forwards and backwards using the 'W' and the 'S' keys
         - Turn left and right by moving the mouse
         - Use the small overhead map and the red arrow to keep track of your position
         - Indicate you are done by pressing the Space bar

   c) Experimenter demonstrates the task of the current scenario
      i) Scenario 1 (abstract 3D environment)
         - Count the number of red objects in the scene or determine pattern formed
         - Move the viewpoint to ensure that all objects have been counted
         - See the effects of X-ray vision turned on versus off
         - Input the number of red objects in the dialog

      ii) Scenario 2 (walkthrough environment)
         - Find the single object in the walkthrough given in the application or count the number of occurrences of this object
         - Explore the building environment using the controls
         - See the effects of X-ray vision turned on versus off
         - Mark the location of the target on the overhead map

   d) Subject is allowed to experiment with the application and ask questions

   e) Move on to tasks when subject indicates readiness

2.3. Task execution (10 minutes per task set) — [repeated for each method]

   a) Experimenter launches the test application for the current scenario and method
      - Scenario 1, X-Ray off: `Game1ReleaseA/noxray.bat`

- Scenario 1, X-Ray on: `Game1ReleaseA/xray.bat`
- Scenario 2, X-Ray off: `Game2ReleaseA/noxray.bat`
- Scenario 2, X-Ray on: `Game2ReleaseA/xray.bat`

b) Experimenter hands over control to subject and informs subject that test will commence when the "Ok" button is pressed

c) Subject solves each task — [repeated for 10 tasks]

- Subject presses the Space bar when done with each task and inputs his or her answer
- Experimenter answers any questions related to technical or study issues (not related to the specific tasks)

3. Post-test questionnaire (5 minutes)

   3.1. Experimenter gives the post-test questionnaire form to the subject

   3.2. Subject fills out the questionnaire

4. Conclusion

   4.1. Experimenter thanks subject for participation in the study

   4.2. Experimenter issues the agreed compensation to the subject (if any)

   4.3. Experimenter asks subject if he or she has any questions and answers them accordingly

   4.4. Subject departs

5. Epilogue

   5.1. Experimenter collects post-test questionnaire

   5.2. Experimenter copies the log files (`stats.dat`) from the specific directories to the results directory

C.2 Scenarios

Scenarios are given in the text of the script in Section C.1 above.

C.3 Post-Test Questionnaire

1. For the abstract environment (counting simple 3D objects in a collection), what did you think of the techniques with regard to the following aspects:

   1.1. Efficiency (how efficient was the technique for solving the task)?

a) Standard
☐ very good ☐ good ☐ neutral ☐ bad ☐ very bad

b) X-Ray
☐ very good ☐ good ☐ neutral ☐ bad ☐ very bad

1.2. Ease (how easy was it to solve the task)?

a) Standard
☐ very good ☐ good ☐ neutral ☐ bad ☐ very bad

b) X-Ray
☐ very good ☐ good ☐ neutral ☐ bad ☐ very bad

1.3. Enjoyability (how enjoyable was it to solve the task?)

a) Standard
☐ very good ☐ good ☐ neutral ☐ bad ☐ very bad

b) X-Ray
☐ very good ☐ good ☐ neutral ☐ bad ☐ very bad

2. For the building environment (finding an object in a simple 3D building), what did you think of the techniques with regard to the following aspects:

2.1. Efficiency (how efficient was the technique for solving the task)?

a) Standard
☐ very good ☐ good ☐ neutral ☐ bad ☐ very bad

b) X-Ray
☐ very good ☐ good ☐ neutral ☐ bad ☐ very bad

2.2. Ease (how easy was it to solve the task)?

a) Standard
☐ very good ☐ good ☐ neutral ☐ bad ☐ very bad

b) X-Ray
☐ very good ☐ good ☐ neutral ☐ bad ☐ very bad

2.3. Enjoyability (how enjoyable was it to solve the task?)

a) Standard
☐ very good ☐ good ☐ neutral ☐ bad ☐ very bad

b) X-Ray
☐ very good ☐ good ☐ neutral ☐ bad ☐ very bad

3. Overall, which vision mode did you prefer?
☐ Standard vision
☐ X-Ray vision

4. Which object is in front of the other? (See Figure C.1.)

4.1. ☐ bike — ☐ white table

4.2. ☐ soda — ☐ director's chair

4.3. ☐ white table — ☐ dog

5. How strong was the depth perception for the X-Ray vision method? (In other words, how well did you feel like you could perceive the depth ordering of objects using the X-Ray vision method?)
☐ very good ☐ good ☐ neutral ☐ bad ☐ very bad

6. Do you have prior experience of either playing 3D computer games (Quake, Half-Life, Unreal, etc) or working with professional 3D software (3D Studio MAX, Milkshape, Maya, etc)? I.e. more than 20 hours combined working with these games/applications?
☐ yes ☐ no

7. In summary, what did you think were the major problems with the two methods you tested (for both environments)? Please give your thoughts and comments.

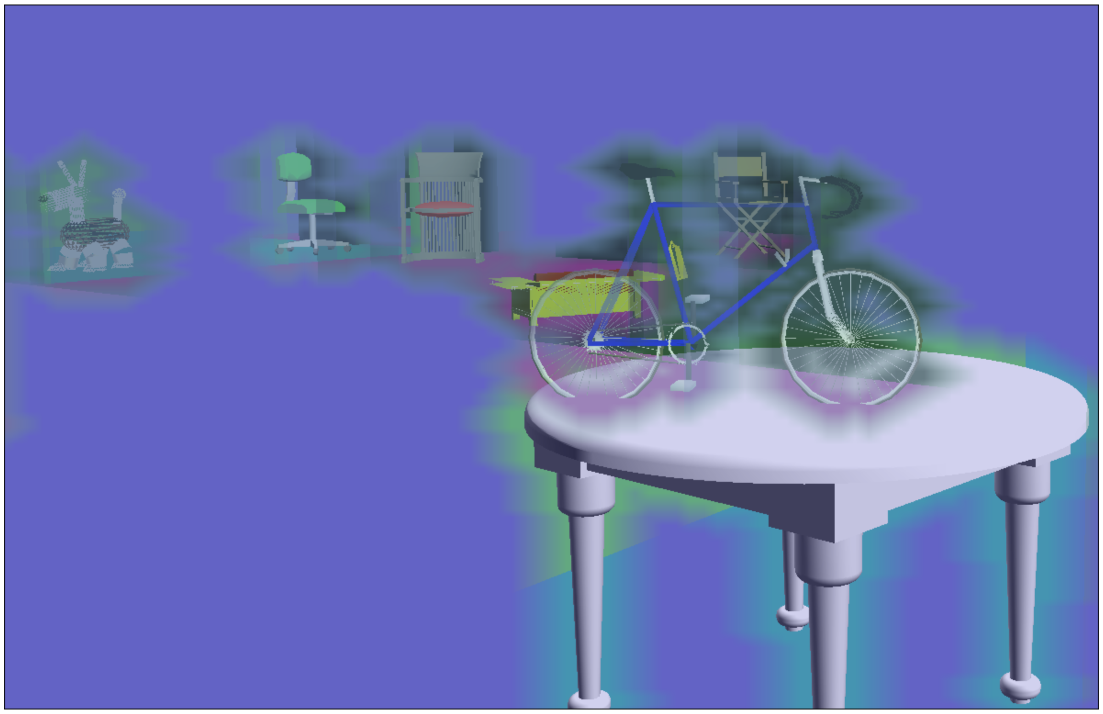


Figure C.1: Depth perception test.

Appendix D

3D Navigation Guidance User Study

D.1 Script

0. Preparations

   0.1. Consent form (2 copies)

   0.2. Post-test questionnaire (1 copy)

   0.3. Experimenter decides which group the subject will belong to

      a) Groups: Free flight, passive follow, or hybrid

   0.4. Experimenter prepares the test scripts with the correct method

      a) Training scenario: `test-training.bat`
      b) Outdoor scenario: `test-outdoor.bat`
      c) Indoor scenario: `test-indoor.bat`
      d) Infoscape scenario: `test-infoscape.bat`
      e) Conetree scenario: `test-conetree.bat`

1. Introduction (5 minutes)

   1.1. Experimenter welcomes subject

   1.2. Experimenter gives the subject the consent form

      a) Experimenter asks the subject to read the consent form
      b) Experimenter asks that the subject should sign the consent form if, and only if, he or she agrees with its contents
      c) Experimenter gives the subject a copy of the consent form for future reference

   1.3. Experimenter gives brief explanation of study

a) Purpose: "The purpose of this study is to explore various ways of navigating in 3D environments. We know that 3D navigation is difficult, especially for large worlds that cannot be seen from a single viewpoint. It is easy to get lost or disoriented, leading to the user not seeing the whole world or missing some of the important targets. Therefore we have developed a number of ways to assist the user in navigating the world."

b) Applications: "Many practical 3D worlds, such as virtual environments, 3D simulations, 3D visualizations, 3D modeling, and even 3D games have this problem. They are typically large enough so that navigation becomes a challenge."

c) Background: "Our method to meet this challenge consists of two steps: a static analysis phase that builds a tour through any kind of 3D world, and an interactive step where the user is guided along the tour. This tour is designed so that it visits all of the important targets in the world. The interactive part of this method is what we will be testing today."

d) Research question: "We want to investigate two separate issues: first, we believe that guiding the user along a tour will increase the efficiency of the user finding the important targets in the world. However, we also believe that having no control over where you are going will reduce you to a passive recipient. Therefore, we also want to investigate whether having a measure of control of your movement will increase your ability to learn a 3D world."

e) Methods: "There are three different navigation methods: (a) free navigation with no assistance, (b) passive navigation with full assistance, and (c) hybrid navigation with partial assistance. We will be testing the navigation methods in four different types of environments: a large outdoor world, a single-level indoor world, an abstract information landscape, and an abstract hierarchy visualization called a conetree. You will only be using one of the three navigation methods for this test."

f) Phases and Tasks: "Each scenario consists of three phases: a familiarization phase, a recall phase, and an evaluation phase. During familiarization, you will be exploring the new world looking for a specific set of three target objects in the world that will be shown to you. After five minutes, this phase is ended. The following recall phase asks you to place two of the three target types on an overhead map of the world. Finally, in the evaluation phase, you will return back to the world looking for the third of the target types. Here, you are supposed to fly up to each of the targets and mark them, making them disappear. When you think you have found all targets of this type, you end the scenario."

g) Measurements: "During the first phase, you will be limited to exploring the world for five minutes. The second recall phase has no time limitation, but both the number of targets of a specific type that you place as well as their distance from their real positions will be recorded. Finally, for the third evaluation phase, a clock will measure the time taken for you to mark targets in the world and then signal that you are done. In addition, the correctness metric, i.e. the number of found targets in comparison to the number of existing ones, is computed and stored. All data will be kept strictly confidential and will be identified with a code number and not your name."

h) Ordering: "The order you receive the scenarios will be randomly determined by the experimenter (me) to avoid effects of practice."

i) Training: "Before starting the experiment itself, you will be given a short training session where you are allowed to familiarize yourself with the navigation method, the three phases, and the tasks that you will be asked to perform in each of them. You may ask questions at any point during this training session. During the test itself, I will only answer questions relating to technical or administrative issues."

j) Experimenter asks whether the subject has any questions and answers them accordingly

1.4. Experimenter determines ordering of scenarios (randomly or by design)

  a) Outdoor, indoor, infoscape, or conetree scenarios

2. Training (5 minutes)

  2.1. Experimenter explains the basic metaphor behind the subject's navigation method

  - Free flight: "In free flight mode, you control the viewpoint as if it was attached to a helicopter flying through the 3D world. You have no access to any navigation assistance method."
  - Passive follow: "In passive follow mode, you are the passenger of a vehicle that flies through the 3D world along a pre-defined tour. The tour has been designed to reveal all of the targets in the world. However, you cannot affect your movement, only look around."
  - Hybrid follow: "In the hybrid follow mode, you are the driver of a vehicle attached to a path going through the world. You can start and stop your vehicle at any time, and also backtrack

along the path. Furthermore, you are able to explore locally in the immediate neighborhood around the path. The path has been designed to reveal all of the targets in the world."

2.2. Experimenter starts the training application

   a) Script: `test-training.bat`

2.3. Experimenter demonstrates the operation of the navigation method and lets the subject try himself or herself

   a) Free flight mode:
      - Change the viewpoint by using the left mouse button and moving the mouse
      - Move forward and backwards along the view direction by using the up and down arrow keys or the 'W' and 'S' keys
      - Slide left or right perpendicular to the view direction by using the left and right arrow keys or the 'A' and 'D' keys

   b) Passive follow mode:
      - Change the viewpoint by using the left mouse button and moving the mouse

   c) Hybrid follow mode:
      - Change the viewpoint by using the left mouse button and moving the mouse
      - Increase forward speed by using the up arrow or 'W' keys
      - Increase backwards movement by using the down arrow or 'S' keys
      - There are three different speeds: forward, stop, or backward
      - The viewpoint is attached to the path as if held by a rubber band
      - Explore forward in the view direction to get a closer look by pressing and holding the center mouse button
      - Explore backward from the view direction to get a wider look by pressing and holding the right mouse button
      - Releasing the right or center buttons will return the view to normal

2.4. Experimenter demonstrates the recall phase and lets the subject try himself or herself

   a) Mark the location of targets of a specific type by clicking and dragging the icon on the map
   b) Move to the next target type by pressing space

2.5. Experimenter demonstrates the evaluation phase and lets the subject try himself or herself

a) Navigate through the world using the "free flight" mode (no assistance)
b) Mark a target by approaching it closely and hitting the Tab button
c) Finish the phase by hitting the Return button

2.6. Experimenter asks whether the subject has any questions and answers them accordingly

3. Scenario execution (10 minutes per scenario) — [repeated for all four scenarios]

3.1. Experimenter informs subject of current scenario and shows the targets

a) Outdoor: "You are a rescue leader with a mission to travel to the site of an accident. Before this, you are given the chance to briefly study the site in a virtual 3D environment. You are specifically interested in the number and position of cars, fire hydrants, and helicopters in the environment."
b) Indoor: "You are a new visitor to a furniture store looking to learn a little about the store before you go exploring yourself. More specifically, you are interested in looking at office chairs, sofas, and floor lamps. You expect to go around in the store after your familiarization, finding what you are looking for in a limited amount of time."
c) Infoscape: "You are planning to write a report for your company. In order to do this, you anticipate making use of a number of Word documents, PDF reports, and some Excel spreadsheets on your computer. Before starting, you will study a 3D version of your filesystem using a special visualization application. You might have to retrieve your source material in very short time when actually writing the report."
d) Conetree: "You are a consultant charged with starting a new project in the company you have been contracted to work for. Before beginning your work, you study the organizational hierarchy of the company using a 3D conetree visualization looking for potential employees with the correct skills. You know you will need to put together a team very quickly, but you are not yet sure exactly which skills are needed for the team members."

3.2. Task execution (8 minutes)

a) Experimenter launches the test application for the current scenario
   - Outdoor scenario: `test-outdoor.bat`
   - Indoor scenario: `test-indoor.bat`
   - Infoscape scenario: `test-infoscape.bat`

- Conetree scenario: `test-conetree.bat`

b) Experimenter initializes the first phase (familiarization) and lets the subject explore the world
- Experimenter announces when half of the time (2.5 minutes) as well as when 1 minute of the time is remaining

c) Experimenter moves to phase two (recall) after 5 minutes of exploration

d) Subject performs the recall phase
- Subject marks the location and number of observed targets of each type
- Subject moves to next type by pressing the Space key

e) Experimenter briefs subject about the third phase
- Free flight for navigating the world
- Mark targets with the Tab key
- End the phase by hitting the Return key

f) Subject performs the evaluation phase and quits the phase when he or she is done

4. Post-test questionnaire (5 minutes)

   4.1. Experimenter gives the post-test questionnaire form to the subject

   4.2. Subject fills out the questionnaire

5. Conclusion

   5.1. Experimenter thanks subject for participation in the study

   5.2. Experimenter issues the agreed compensation to the subject (if any)

   5.3. Experimenter asks subject if he or she has any questions and answers them accordingly

   5.4. Subject departs

6. Epilogue

   6.1. Experimenter collects post-test questionnaire

   6.2. Experimenter copies the log files (*data.dat*) from the specific directories to the results directory

D.2 Scenarios

Scenarios are given in the text of the script in Section D.1 above.

D.3 Post-Test Questionnaire

1. It was easy solving the tasks I was asked to perform.

   1.1. Indoors
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

   1.2. Outdoors
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

   1.3. Infoscape
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

   1.4. Conetree
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

2. The navigation method helped me solve my tasks efficiently.

   2.1. Indoors
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

   2.2. Outdoors
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

   2.3. Infoscape
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

   2.4. Conetree
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

3. I enjoyed solving the tasks using the navigation method.

   3.1. Indoors
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

   3.2. Outdoors
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

   3.3. Infoscape
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

   3.4. Conetree
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

4. I never felt lost when moving around using the navigation method.

   4.1. Indoors
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

   4.2. Outdoors
   ☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

4.3. Infoscape
☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

4.4. Conetree
☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

5. I was confident marking objects in the second (recall) phase.

5.1. Indoors
☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

5.2. Outdoors
☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

5.3. Infoscape
☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

5.4. Conetree
☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

6. I was confident finding targets in the third (evaluation) phase.

6.1. Indoors
☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

6.2. Outdoors
☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

6.3. Infoscape
☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

6.4. Conetree
☐ fully agree ☐ agree ☐ neutral ☐ disagree ☐ fully disagree

7. Do you have prior experience of either playing 3D computer games (Quake, Half-Life, Unreal, etc) or working with professional 3D software (3D Studio MAX, Milkshape, Maya, etc)? I.e. more than 20 hours combined working with these games/applications?
☐ yes ☐ no

8. In summary, what did you think were the major problems with the navigation method you tested (for all worlds)? Please give your thoughts and comments.

Appendix E

Growing Squares User Study

E.1 Script

0. Preparations
    - 0.1. Task scenario cards (4 copies)
1. Introduction (5 minutes)
    - 1.1. Experimenter welcomes subject
    - 1.2. Experimenter gives brief explanation of study
        - a) Purpose: comparative evaluation of two different ways of visualizing causal relations
        - b) Old way versus a new information visualization technique
        - c) Each subject will use the two technique for two different information densities
        - d) Hardware: Pentium III laptop with 3D accelerator
        - e) Software: information visualization application (CausalViz) running on the Linux operating system
        - f) Study data will be reported anonymously in a Ph.D. thesis and possibly in an academic paper
2. Training (5 minutes)
    - 2.1. Experimenter demonstrates CausalViz's user interface
        - a) Main window used for controlling the visualization
        - b) Animation controls and timeline for selecting positions in time
        - c) Visualization windows for different views of the same data
        - d) Zoomable interface: right-click and move to zoom, left-click and move to pan

2.2. Experimenter explains the information visualization technique relevant to the subject

   a) Hasse diagrams is a traditional technique (animated). Processes as swimlanes, messages as arrows between them.
   b) 2D squares is a new visualization technique. Processes as growing squares, the color signifying influences from other processes at different points in time.
   c) Causal relation: process A has received message from process B at time $t_0 \Rightarrow A$ causally related to B for all times $t > t_0$.

2.3. Experimenter demonstrates how to solve common problems using both visualizations

   a) Duration comparison: find the processes that have the longest life time
   b) Causal relation: find the processes that have influenced a given target process

2.4. Subject is allowed up to 5 minutes of practice using a test scenario

   a) Local file: `practice.xml`
   b) Subject decides when ready to proceed

3. Task explanation (5 minutes)

3.1. Subject will use both the old and new visualization to allow for comparison

3.2. Subject will work with two different data sets of increasing density for each visualization

3.3. All in all, the subject will work with four different tasks for four different data sets

3.4. Subject is given a written booklet of subtasks for each data set

   a) Each subtask is a common activity used in reality
   b) Use application visualization to find out solutions to task
   c) It is okay to skip or guess the solution of a subtask

3.5. Subject may use additional paper for sketching and quick notes

3.6. Subtasks will be timed by the experimenter

   a) The idea is to work fast without being sloppy
   b) Better to be correct than to be fast
   c) Do not go back to previous subtasks
   d) Wait until indicated before proceeding to a new subtask

3.7. Subject will respond to a short questionnaire at the end of each task

3.8. Subject will respond to a final questionnaire at the end of the test

3.9. Questions to the experimenter are OK at any time

4. Task execution (5-10 minutes per task) — [repeated for both techniques and densities]

   4.1. Data density
      a) Sparse: 10 processes, 30 messages
      b) Dense: 30 processes, 90 messages

   4.2. Experimenter loads the data sets
      a) Local file (sparse, old): `sparse1.xml`
      b) Local file (sparse, new): `sparse2.xml`
      c) Local file (dense, old): `dense1.xml`
      d) Local file (dense, new): `dense2.xml`

   4.3. Experimenter closes down windows of visualizations that should not be active

   4.4. Subject is allowed to proceed with solving the task and write down the solutions
      a) Experimenter will time each subtask and later check performance
      b) Experimenter randomly generates 2 questions specific to the data set (see problem sheet)

   4.5. Subject is asked to respond to the questionnaire for the specific task performed

5. Post-test questionnaire (5 minutes)

   5.1. Experimenter gives the post-test questionnaire form to the subject

   5.2. Subject fills out the questionnaire

6. Conclusion

   6.1. Experimenter thanks subject for participation in the study

   6.2. Experimenter issues the agreed compensation to the subject (if any)

   6.3. Experimenter asks subject if he or she has any questions and answers them accordingly

   6.4. Subject departs

E.2 Scenarios

E.2.1 Duration Comparison

You are analyzing the given distributed system to find out which processes take the most CPU time in the system. In order to do this, find the process that has the longest duration in the sequence (from start to finish).

E.2.2 Influence Importance

You now want to know which are the most important processes in the system. One way to figure this out is to see which processes influenced the most other processes (directly or indirectly). Find the process that has had the most influence on the system.

E.2.3 Influence Assessment

Some of the processes in the system are clearly "worker" processes or servers, which receive commands from other processes and in exchange perform some work. Identify these server processes by finding the process that was influenced by the most other processes (directly or indirectly).

E.2.4 Inter-Node Causal Relations

In order to ensure that the system is behaving properly, you want to check that some node x has been influenced by another node y. Answer the following questions (remember, causal relations are transitive, so check indirect relations too):

1. Is process _____ causally related to (influenced by) _____? ☐ yes ☐ no
2. Is process _____ causally related to (influenced by) _____? ☐ yes ☐ no
3. Is process _____ causally related to (influenced by) _____? ☐ yes ☐ no

E.3 Post-Task Questionnaire

1. Please rate the visualization system according to ease of use.
   ☐ very hard
   ☐ hard
   ☐ medium
   ☐ easy
   ☐ very easy

2. Please rate the visualization system according to efficiency.
   ☐ very inefficient
   ☐ inefficient
   ☐ neutral
   ☐ efficient
   ☐ very efficient

3. Please rate the visualization system according to enjoyment.
   ☐ very unpleasant
   ☐ unpleasant
   ☐ neutral
   ☐ pleasant
   ☐ very pleasant

E.4 Post-Test Questionnaire

1. Please select the visualization system that you liked more with respect to ease of use.
   ☐ Hasse Diagrams
   ☐ Growing Squares

2. Please select the visualization system that you liked more with respect to efficiency.
   ☐ Hasse Diagrams
   ☐ Growing Squares

3. Please select the visualization system that you liked more with respect to enjoyment.
   ☐ Hasse Diagrams
   ☐ Growing Squares

Appendix F

Growing Polygons User Study

F.1 Script

0. Preparations
   - 0.1. Task scenario cards (4 copies)
1. Introduction (5 minutes)
   - 1.1. Experimenter welcomes subject
   - 1.2. Experimenter gives brief explanation of study
     - a) Purpose: comparative evaluation of two different ways of visualizing causal relations
     - b) Old way versus a new information visualization technique
     - c) Each subject will use the two technique for two different information densities
     - d) Hardware: Pentium III laptop with 3D accelerator
     - e) Software: information visualization application (CausalViz) running on the Linux operating system
     - f) Study data will be reported anonymously in a Ph.D. thesis and possibly in an academic paper
2. Training (5 minutes)
   - 2.1. Experimenter demonstrates CausalViz's user interface
     - a) Main window used for controlling the visualization
     - b) Animation controls and timeline for selecting positions in time
     - c) Visualization windows for different views of the same data
     - d) Zoomable interface: right-click and move to zoom, left-click and move to pan

2.2. Experimenter explains the information visualization technique relevant to the subject

a) Hasse diagrams is a traditional technique (animated). Processes as swimlanes, messages as arrows between them.

b) 2D polygons is the new technique with colors used to signify process influences. Age rings on a tree, grows outwards. Sectors signifying different processes.

c) Use of the mini-map for navigating the visualization

d) Use of the animation toolbar

e) Causal relation: process A has received message from process B at time $t_0 \Rightarrow A$ causally related to B for all times $t > t_0$.

2.3. Experimenter demonstrates how to solve common problems using both visualizations

a) Duration comparison: find the processes that have the longest life time.

b) Influence dominance: find the process that has the most influence in the system.

c) Influence assessment: find the process that has been influenced the most in the system.

d) Causal relation: find the processes that have influenced a given target process.

2.4. Subject is allowed up to 5 minutes of practice using a test scenario

a) Local file: `practice.xml`

b) Subject decides when ready to proceed

3. Task explanation (5 minutes)

3.1. Subject will use both the old and new visualization to allow for comparison

3.2. Subject will work with two different data sets of increasing density for each visualization

3.3. All in all, the subject will work with four different tasks for four different data sets

3.4. Subject is given a written booklet of subtasks for each data set

a) Each subtask is a common activity used in reality

b) Use application visualization to find out solutions to task

c) It is okay to skip or guess the solution of a subtask

3.5. Subject may use additional paper for sketching and quick notes

3.6. Subtasks will be timed by the experimenter
   a) The idea is to work fast without being sloppy
   b) Better to be correct than to be fast
   c) Do not go back to previous subtasks
   d) Wait until indicated before proceeding to a new subtask

3.7. Subject will respond to a short questionnaire at the end of each task

3.8. Subject will respond to a final questionnaire at the end of the test

3.9. Questions to the experimenter are OK at any time

4. Task execution (5-10 minutes per task) — [repeated for both visualizations and densities]

   4.1. Data density
      a) Sparse: 5 processes, 15 messages
      b) Dense: 20 processes, 60 messages

   4.2. Experimenter loads the data sets
      a) Local file (sparse, old): `sparse3.xml`
      b) Local file (sparse, new): `sparse4.xml`
      c) Local file (dense, old): `dense3.xml`
      d) Local file (dense, new): `dense4.xml`

   4.3. Experimenter closes down windows of visualizations that should not be active

   4.4. Subject is allowed to proceed with solving the task and write down the solutions
      a) Experimenter will time each subtask and later correct performance
      b) Experimenter randomly generates 2 questions specific to the data set (see problem sheet)

   4.5. Subject is asked to respond to the questionnaire for the specific task performed

5. Post-test questionnaire (5 minutes)

   5.1. Experimenter gives the post-test questionnaire form to the subject

   5.2. Subject fills out the questionnaire

6. Conclusion

   6.1. Experimenter thanks subject for participation in the study

   6.2. Experimenter issues the agreed compensation to the subject (if any)

6.3. Experimenter asks subject if he or she has any questions and answers them accordingly

6.4. Subject departs

F.2 Scenarios

F.2.1 Duration Comparison

You are analyzing the given distributed system to find out which processes take the most CPU time in the system. In order to do this, find the process that has the longest duration in the sequence (from start to finish).

F.2.2 Influence Importance

You now want to know which are the most important processes in the system. One way to figure this out is to see which processes influenced the most other processes (directly or indirectly). Find the process that has had the most influence on the system.

F.2.3 Influence Assessment

Some of the processes in the system are clearly "worker" processes or servers, which receive commands from other processes and in exchange perform some work. Identify these server processes by finding the process that was influenced by the most other processes (directly or indirectly).

F.2.4 Inter-Node Causal Relations

In order to ensure that the system is behaving properly, you want to check that some node x has been influenced by another node y. Answer the following questions (remember, causal relations are transitive, so check indirect relations too):

1. Is process ____ causally related to (influenced by) ____? ☐ yes ☐ no
2. Is process ____ causally related to (influenced by) ____? ☐ yes ☐ no
3. Is process ____ causally related to (influenced by) ____? ☐ yes ☐ no

F.3 Post-Task Questionnaire

1. Please rate the visualization system according to ease of use.
   ☐ very hard

☐ hard
☐ medium
☐ easy
☐ very easy

2. Please rate the visualization system according to efficiency.
   ☐ very inefficient
   ☐ inefficient
   ☐ neutral
   ☐ efficient
   ☐ very efficient

3. Please rate the visualization system according to enjoyment.
   ☐ very unpleasant
   ☐ unpleasant
   ☐ neutral
   ☐ pleasant
   ☐ very pleasant

F.4 Post-Test Questionnaire

1. Which of the two visualization techniques did you like more with respect to ease of use?

   ☐ Hasse Diagrams
   ☐ Growing Polygons
   ☐ Undecided

2. Please select the visualization system that you liked more with respect to efficiency.

   2.1. Duration comparison (lifetime of processes)
   ☐ Hasse diagrams
   ☐ Growing Polygons
   ☐ Undecided

   2.2. Influence importance (most influential process)
   ☐ Hasse diagrams
   ☐ Growing Polygons
   ☐ Undecided

   2.3. Influence assessment (most influenced process)
   ☐ Hasse diagrams
   ☐ Growing Polygons
   ☐ Undecided

2.4. Inter-node causal relations (finding causal relations between processes)
- ☐ Hasse diagrams
- ☐ Growing Polygons
- ☐ Undecided

3. Which of the two visualization techniques did you like more with respect to enjoyment?
   - ☐ Hasse diagrams
   - ☐ Growing Polygons
   - ☐ Undecided

Appendix G

CiteWiz User Study

G.1 Script

0. Preparations
    - 0.1. Task scenario booklet (1 copy)
1. Introduction (5 minutes)
    - 1.1. Experimenter welcomes subject
    - 1.2. Experimenter gives brief explanation of study
        - a) Purpose: comparative evaluation of two different ways of visualizing scientific citation networks
        - b) Old way using a database search interface versus a new information visualization technique
        - c) Each subject will use one of the two techniques to solve a set of task scenarios (between-subjects)
        - d) Hardware: Pentium III desktop computer with 3D accelerator
        - e) Software: information visualization application (CiteWiz) running on the Linux operating system
        - f) Study data will be reported anonymously in a Ph.D. thesis and possibly in an academic paper
    - 1.3. Experimenter gives brief background on scientific citation networks
        - a) Scientific work builds on the work conducted by others
        - b) Previous that has influenced the author(s) are mentioned in the references of a scientific paper
        - c) Scientific papers and their references (citations) together make up a large directed graph showing the influences and dependencies of the scientific work in the field

d) Studying the citation network can give us interesting information, such as identifying important papers, successful researchers, and hot topics within the field

2. Training (5 minutes)

   2.1. Experimenter demonstrates the interface of the method the subject will use

   2.2. Database interface

      a) Searching – dialog for free search
      b) Sorting – sorting of the current result set
      c) Filtering – filtering the current result set according to some criteria (based on free text in the specific fields)
      d) Details-on-demand – calling up details on a specific paper

   2.3. CiteWiz visualizations

      a) Main window with the citation database and the view
      b) Construction of the view, adding entries, building groups, nesting groups
      c) Creating a visualization from the current view
      d) Interacting with the CiteWiz GP visualization
         i) Zoomable interface: right-click and move to zoom, left-click and move to pan
         ii) Sliders for linear time windows – segments, window width, and position sliders
         iii) Idea behind GP visualization technique – influences, colors, and polygons
         iv) "Horns" showing forward influences, sectors showing backwards influences
         v) Interaction techniques – collapsing, expanding and selecting a node in the view hierarchy
         vi) Legend window – navigation and color mapping
         vii) Tree manager window – collapsing, expanding, and searching in the view hierarchy
      e) Interacting with the CiteWiz Newton's Shoulders diagrams for both authors and articles

   2.4. Experimenter demonstrates how to solve common problems using both methods

      a) Paper retrieval: finding a specific paper in the citation database
      b) Find related papers: following references backwards (and forward)

    c) Study author collaboration: compare collaboration between two authors

  2.5. Subject is allowed up to 5 minutes of practice using a test scenario

    a) Local file: `iv04dataset.xml`
    b) Subject decides when ready to proceed

3. Task explanation (5 minutes)

  3.1. Subject will use one of the two methods

  3.2. Subject is given a booklet of subtasks for each data set

    a) Each subtask is a common activity used in reality
    b) Use application visualization to find out solutions to task
    c) It is okay to skip or guess the solution of a subtask

  3.3. Subject may use additional paper for sketching and quick notes

  3.4. Subtasks will be timed by the experimenter

    a) The idea is to work fast without being sloppy
    b) Better to be correct than to be fast
    c) Do not go back to previous subtasks
    d) Wait until indicated before proceeding to a new subtask

  3.5. Subject will respond to a short questionnaire at the end of the test

  3.6. Questions to the experimenter are OK at any time

4. Task execution (5-10 minutes per task) — [repeated for each task]

  4.1. Experimenter loads the data set

    a) Local file: `infovis-dataset.xml`

  4.2. Subject is allowed to proceed with solving the task and writing down the solutions

    a) Experimenter will time each subtask and later correct performance

  4.3. Maximum times for each task is 15 minutes

5. Post-test questionnaire (5 minutes)

  5.1. Experimenter gives the post-test questionnaire form to the subject

  5.2. Subject fills out the questionnaire

6. Conclusion

  6.1. Experimenter thanks subject for participation in the study

6.2. Experimenter issues the agreed compensation to the subject (if any)

6.3. Experimenter asks subject if he or she has any questions and answers them accordingly

6.4. Subject departs

G.2 Scenarios

G.2.1 Find a Paper

You need to locate three specific papers. Please use the search feature of the tool to find these papers using the specified search terms. Please write down the full title of each paper.

G.2.2 Find the Most Influential Paper

You are trying to identify the most influential papers of the IEEE Information Visualization conferences. Use the search, browse and visualization facilities of the tool to identify the full title and author of the most influential paper for a specified year of the conference.

G.2.3 Study Author Collaboration

You are studying the collaboration between different authors active in the IEEE InfoVis conferences. Study the relations between authors X and Y. Identify the set of authors that have co-authored papers with both X and Y.

G.3 Post-Test Questionnaire

1. Rate the tool with respect to ease of use.

   ☐ very hard
   ☐ hard
   ☐ medium
   ☐ easy
   ☐ very easy

2. Rate the tool with respect to efficiency of solving the different tasks.

   2.1. Find a paper

   ☐ very inefficient
   ☐ inefficient
   ☐ neutral

☐ efficient
☐ very efficient

2.2. Find the most influential paper

☐ very inefficient
☐ inefficient
☐ neutral
☐ efficient
☐ very efficient

2.3. Study author collaboration

☐ very inefficient
☐ inefficient
☐ neutral
☐ efficient
☐ very efficient

3. Rate the tool with respect to enjoyment.

☐ very unpleasant
☐ unpleasant
☐ neutral
☐ pleasant
☐ very pleasant