AR Magic Lenses: Addressing the Challenge of Focus and Context in Augmented Reality

---

A thesis
submitted in partial fulfilment
of the requirements for the Degree
of
Doctor of Philosophy
in the
University of Canterbury
by
Julian Looser

Examining Committee

Associate Professor Andy Cockburn	Supervisor
Professor Mark Billinghurst	Co-Supervisor
Professor Bruce Thomas	Examiner
Professor Hans Gellersen	Examiner

University of Canterbury
2007

Declaration

All contributions described in this thesis are entirely my own, with the exception of cases where I collaborated with other researchers. The following list identifies such parts of my research, and clarifies the roles of the contributors.

Chapter 5 - Stencil Lenses: The work on stencil-based AR Magic Lenses was primarily my own. I developed the software, explored the application space, and wrote the majority of the paper describing the work. The publication, presented at GRAPHITE 04, was written with the assistance of my supervisors, Mark Billinghurst and Andy Cockburn.

Chapter 5 - TankWar: The Augmented Reality game, TankWar, described in this chapter was the idea of Trond Nilsen, and became the topic of his Masters research (Nilsen 2006). My contribution to TankWar was software development of the first OpenGL version, which included the AR Magic Lens interaction techniques and a transitional (AR-VR) interface. In addition, I provided technical advice on the development of Trond's second iteration of TankWar, built with Open Scene Graph. I co-authored a paper about TankWar with Trond and Steven Linton (Nilsen, Linton and Looser 2004), and helped demonstrate TankWar at the New Zealand Game Developers Conference in 2004.

Chapter 6 - osgART: The software library, osgART, was originally my invention, as I was individually responsible for developing the first version. osgART has since become a collaborative project, being actively maintained and improved by myself, Hartmut Seichter, Raphaël Grasset and Phil Lamb. osgART has been released under GPL and commercial licenses, and was presented at the Industrial Workshop at ISMAR 06 (Looser, Grasset, Seichter and Billinghurst 2006).

Chapter 7 - AR Selection Experiment: The experiment evaluating AR Magic Lenses as a selection technique was designed and run by myself. The resulting publication, presented at GRAPHITE 07, was written with the assistance of Raphaël Grasset and Mark Billinghurst.

Chapter 8 - Flexible Sheet Lenses: The invention of flexible sheet lenses is my own, and I wrote the software implementation. This work was exploratory and I had many discussions about potential applications and interaction techniques with Raphaël Grasset. Our publication about the flexible lens, presented by Raphaël at ISMAR 07, was written with the additional assistance of Mark Billinghurst.

Dedicated to
Alan and Frieda

Abstract

In recent years, technical advances in the field of Augmented Reality (AR), coupled with the acceleration in computer and graphics processing power, have brought robust and affordable AR within the reach of the wider research community. While the technical issues of AR remain heavily researched, there is also a growing amount of work on user interface development and evaluation, heralding the convergence of traditional Human Computer Interaction (HCI) and AR.

Magic Lenses are 2D interface components that provide alternative representations of objects seen through them. In this way, they can be used to provide Focus and Context in the interface, especially when visualising layered information. There are very few, if any, formal evaluations to guide the development of lens-based interfaces.

This thesis describes the development and evaluation of Magic Lenses as a tool for AR interfaces. The work starts with a comprehensive survey of many Focus and Context techniques, which are classified based on the way they present views to the users for example, a Magic Lens is a spatially separated multiple view technique. A formal evaluation of 2D Magic Lenses in a GIS scenario found that users strongly preferred the lens-based interaction technique to others, largely because it reduced the effort of interaction. Accuracy was high with the lenses, but a simple "global view" interface allowed significantly faster performance.

This positive result motivated further work on Magic Lenses within AR, where the lens metaphor can reinforce the tangible interaction methods that link virtual and real content. To support rapid exploration of interaction alternatives with AR Magic Lenses, I describe the design and architecture of osgART, an AR development toolkit that is available to the research community as open-source software.

Object selection and manipulation is a fundamental interaction requirement for all AR interfaces, and I establish an empirical foundation of performance in this task with a variety of AR interaction techniques, including Magic Lenses. Results show that performance with all techniques is successfully modelled by Fitts' Law, and that Magic Lenses outperformed other

techniques.

Finally, I examine new interaction techniques based on Magic Lenses, particularly a Flexible Sheet Lens, which allows concurrent bimanual specification of multiple parameters within the visualisation.

Table of Contents

List of Figures

List of Tables

Acknowledgments

This thesis would not have been possible without the help and support of a great many people. I would like to take the time to thank them here.

Firstly, I must thank my supervisor, Dr Andy Cockburn, for his constant support, wisdom and enthusiasm throughout my studies. It was Andy's suggestion to pursue the PhD in the beginning, and his belief in me has borne me through.

Immense thanks are also due to Dr Mark Billinghurst, my co-supervisor. Without Mark the HIT Lab would not exist at Canterbury University, and I would never have had the option to research augmented reality here. The HIT Lab has been a wonderful research environment, and has provided me with countless new opportunities and experiences.

HIT Lab NZ would also not exist without the vision of Dr Tom Furness, director of the original HIT Lab in Seattle. Tom hosted me for a five month internship within the HIT Lab at the University of Washington. It was a great and memorable experience for which I am truly thankful.

I would also like to thank my thesis examiners, Professors Bruce Thomas and Hans Gellersen, for taking the time to read, consider and evaluate my work.

My family has provided constant practical and emotional support. Frieda, Joan, Diana, Colin, Andrei - thank you for all your help, encouragement, patience and love.

The post-doc researchers at HIT Lab NZ provided invaluable advice and assistance. Raphaël, Hartmut and Andreas - thank you for always making time to talk.

I have also benefited from good friends to help and support me. Thanks to Josh, Catherine, Trond, Jörg, Chandra, Matt, Phil, Minkyung, Christiaan, Andy and Nathan... and all the other staff and students of HIT Lab NZ.

Chapter 1

Introduction

1.1 Overview of Thesis and Research Approach

The topic of this thesis is the development and evaluation of Magic Lenses as a Focus and Context technique in Augmented Reality (AR). This research lies at the intersection of two expansive areas of Human Computer Interaction (HCI): Focus and Context techniques, of which Magic Lenses are a specific example, and Augmented Reality, a technology that merges virtual images with the real world. These are both considerably broad fields that are heavily and actively researched, albeit traditionally by separate communities.

Research into AR has historically concentrated on the technical issues of accurately placing virtual information within the user's environment. Gradually these challenges have been overcome or rendered inconsequential by increasingly powerful computer hardware, advanced computer-vision algorithms and higher-resolution trackers. AR development is now within the scope of the average programmer, and attention is turning to the issues of user interface design and visualisation, indicating the approaching convergence of AR and Focus and Context techniques.

Focus and Context is the umbrella term for the challenges and solutions encountered when dealing with extensive data sets where one needs to examine some parts closely and at the same time maintain sufficient awareness of peripheral information. This situation arises frequently in graphical user interfaces where the constraints of screen space and display resolution make it difficult to present large information spaces effectively.

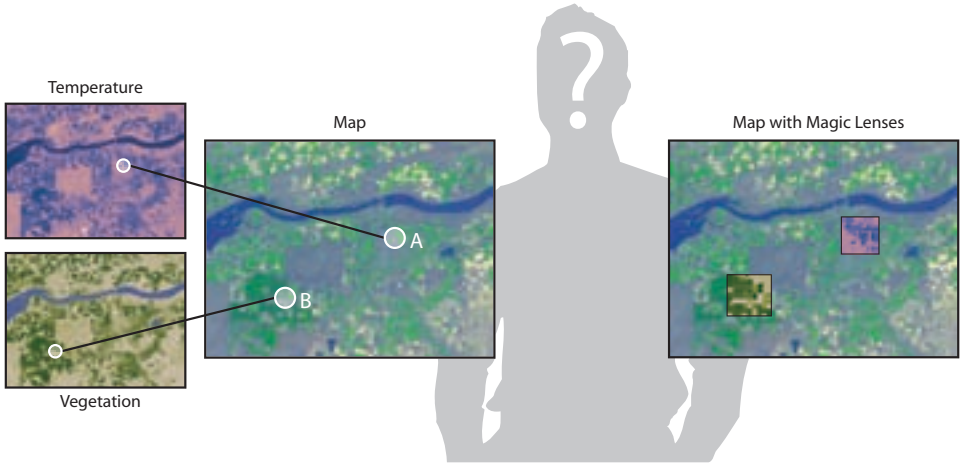


Figure 1.1: Magic Lenses can be used to efficiently visualise layered information.

Magic Lenses are visual filters that display different representations of information. With consideration to the challenge of providing Focus and Context, Magic Lenses can be particularly useful tools for analysing layers of information and when following multiple concurrent paths of analysis. For example, how does the temperature at location A correlate with vegetation at location B? Two visualisation strategies for answering this type of question are shown in Figure 1.1.

Augmented Reality is a display technology that enhances the real world through the real time overlay of virtual information onto real objects and surroundings. Augmented Reality offers new ways to display the visual output of a computer program as well as providing new ways to interact with that program.

This thesis is concerned with the interplay of the above concepts. The specific goals of this thesis are:

- To increase knowledge and awareness of Magic Lenses as user interface tools, through survey, distillation, and empirical evaluation.

- To develop and explore new interaction styles for Augmented Reality

interfaces, based on Magic Lenses, and evaluate their effectiveness.

1.2 Chapter Summary

This section provides a road-map of the chapters in this thesis.

Chapter 2 Related Work presents related work in the areas of Focus and Context, Magic Lenses and Augmented Reality. A survey of Focus and Context techniques is provided, guided by a taxonomy developed to give structure to the survey, and to demonstrate how Focus and Context techniques relate to Graphical User Interfaces in general. The taxonomy first draws a distinction between techniques that use a single view and those that combine two or more views. Single view techniques are divided into two types: Static, which cleverly present detailed and contextual information at the same time, and Dynamic, which transition between levels of detail, presenting a complete picture of the information over time. Multiple view techniques are also divided in a similar way. Spatial multiple-view techniques share screen space between views, but display them at the same time. Temporal multiple-view techniques display views one at a time, but each view occupies the entire workspace while active. Static, Dynamic, Spatial and Temporal techniques are all described with examples from the literature.

Magic Lenses are identified as a particular type of Spatial multiple-view technique. A full survey of prior work involving Magic Lenses in 2D user interfaces is provided, followed by a similar survey of work where the Magic Lens has been applied in 3D virtual environments.

This leads to the topic of Augmented Reality, a particular type of 3D virtual environment that overlays computer graphics on the real world. A solid foundation in this topic is provided, including the process of providing an AR experience, the role of usability testing, and a collection of examples of AR in various domains.

Chapter 3 Fundamentals of Magic Lenses delves deeper into the topic of 2D Magic Lenses. The properties and capabilities that make Magic Lenses valuable user interface tools are explored.

Chapter 4 2D Magic Lens Evaluation reports on a formal evaluation that examines 2D Magic Lenses as information filtering tools in a mapping scenario.

Chapter 5 Augmented Reality Magic Lenses introduces AR Magic Lenses, the core innovation within this thesis. AR Magic Lenses apply the concepts of traditional Magic Lens filters to the compelling domain of Augmented Reality, where the boundary between the digital and real world is blurred.

Chapter 6 osgART describes the design and implementation of osgART, a software library that integrates video sources, vision-based tracking libraries, and the Open Scene Graph rendering framework. osgART was developed to enable AR Magic Lens concepts to be rapidly prototyped, but can be used to quickly create other AR applications as well.

Chapter 7 AR Magic Lens Selection Evaluation reports on the formal evaluation of AR Magic Lenses as a tool for the crucial task of object selection in 3D virtual environments. The Magic Lens is compared to the traditional selection approaches of Direct Touch and Ray-Casting.

Chapter 8 Flexible Sheet Lenses extends the development of AR Magic Lenses by moving beyond the magnifying glass metaphor. New modes of interaction are explored by abandoning the circular lens in favour of a flexible virtual sheet. The sheet is manipulated with two physical handles, introducing interesting possibilities based on bimanual interaction and gesture recognition.

Chapter 9 Discussion and Future Work presents a discussion of what has been learned throughout the thesis, how it relates to other research, and proposes directions for future work.

Chapter 10 Conclusion provides a concise summary of the contributions of this thesis to the collective knowledge and understanding surrounding

Magic Lenses, particularly in Augmented Reality.

1.3 Contributions

The main contributions of this thesis are:

- A taxonomy of Focus and Context techniques.
- A formal evaluation of 2D Magic Lenses for information analysis and filtering.
- The merger of the concepts of Magic Lenses and Tangible Augmented Reality to produce AR Magic Lenses.
- The design and implementation of a software library, osgART, to expedite the development of Augmented Reality applications.
- A formal evaluation of AR Magic Lenses for object selection.
- The invention and implementation of the Flexible Sheet Lens, and the exploration of new interaction techniques it makes possible.

Chapter 2

Related Work

This chapter presents surveys of work in the areas of Focus and Context and Augmented Reality, followed by a detailed description of Magic Lenses.

2.1 Focus and Context

2.1.1 An Overview of Graphical User Interfaces

The Graphical User Interface, or GUI, is currently the primary means of interacting with computers. GUIs were heavily developed in the late seventies and early eighties (the Xerox Alto is a notable example, having a GUI in 1973), and as Figure 2.1 shows, the essential elements of desktop environments have changed very little since then. All computer users are familiar with this ubiquitous desktop interface, in which our workspaces are populated by windows, icons, menus and pointers: the WIMP interaction style.

Using the keyboard and mouse to interact with visual widgets such as buttons, scrollbars, textboxes and sliders is the established norm for interacting with computers. This style of input was described by Shneiderman (1987) as *direct manipulation*. The underlying motivation for direct manipulation is the premise that users will find tasks easier to perform when presented with an interface that employs familiar metaphors, and maps closely to their own mental model. For example, users understand from real-world experience that a physical button activates something when pushed. Therefore, it is theoretically only a small cognitive leap to understand a virtual button on a computer screen. The concept of direct manipulation is the foundation of today's user interfaces. Apple and Microsoft both support this approach in their user interface development guidelines (Apple Computer 1992, Microsoft Corporation 1999).

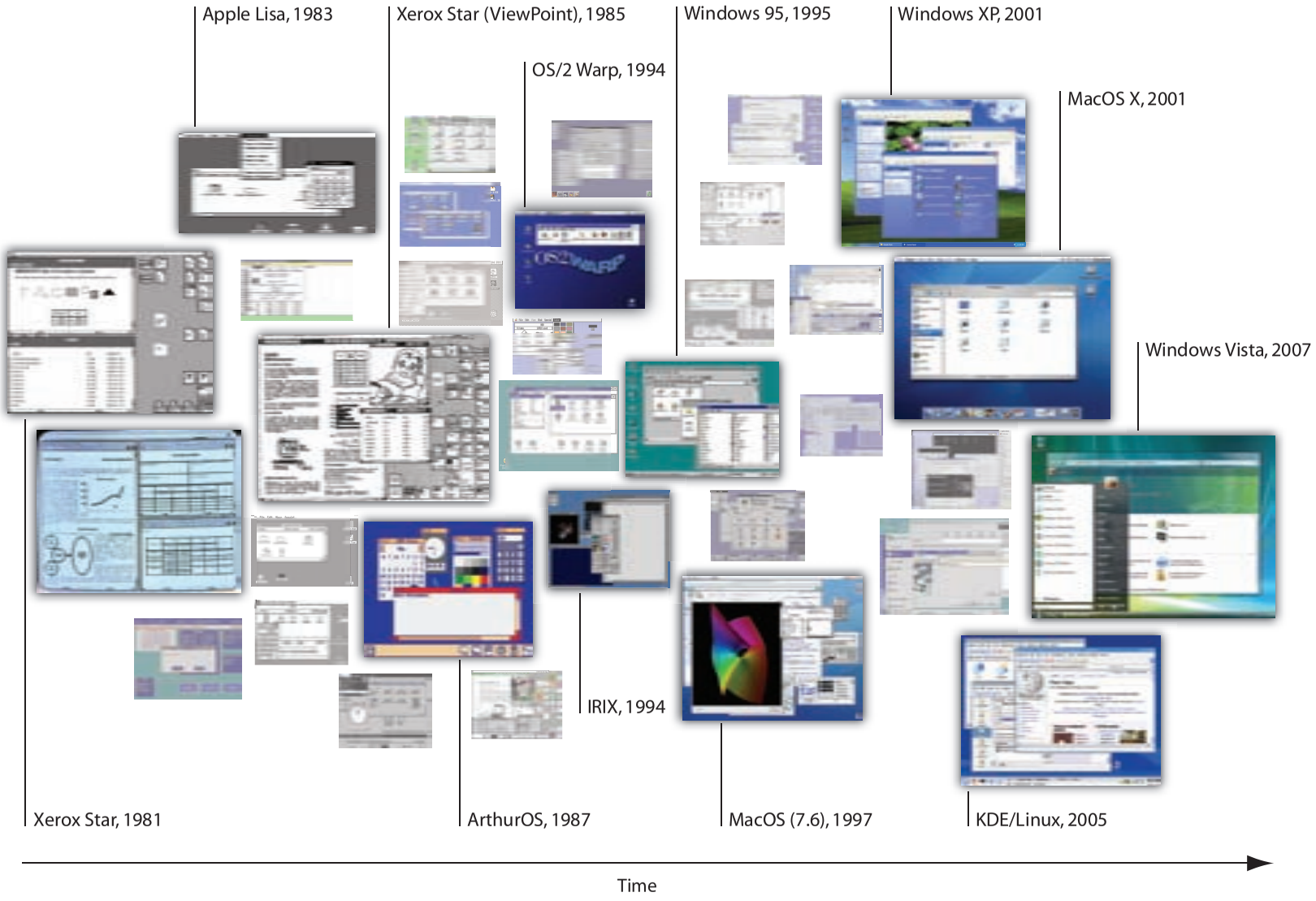


Figure 2.1: WIMP-based operating systems from the early 1980s to the present day. This figure illustrates the similar appearance and themes that have remained in place over time.

Computer users must frequently deal with large documents, images and diagrams. Software that operates on these objects must create views that facilitate the tasks users want to achieve, such as editing, searching and browsing. An unavoidable constraint that affects the generation of such views is the physical limitations of the display device.

Over the last few decades computer displays have improved significantly in terms of resolution, bit-depth, brightness, contrast and form-factor. In the early 1980s it was common to use a CRT (Cathode Ray Tube) monitor with a resolution of only 320x200 pixels and support for four colours. In 2007 it is common to use an LCD (Liquid Crystal Display) with a resolution of 1280x1024 pixels and many millions of colours. Furthermore, many users are embracing multiple-monitor configurations to increase their display space.

Although this upward trend shows no sign of slowing down, the quantity of information our computers can access, store and process has increased at a much higher rate. The outcome is that we have an ever-increasing amount of information available and finite screen spaces on which to display it. The screen is our window into the information space and it has become the bottleneck.

Although information spaces can be immense, users generally have a particular *region of interest* that pertains to their current task. For example, when editing a large document in a word processor, the current page, paragraph, sentence or word may be the region of interest, depending on the user's specific objective. Displaying large information spaces such that the region of interest appears at a comfortable size often means that most of the content falls outside the screen area. Conversely, at a scale where everything fits on the screen at once, the region of interest can become too small, cluttered or occluded by other information to be seen clearly. Neither of these conditions is ideal as the user is forced to choose between views that provide local detail or a global overview, when in fact they require a measure of both to work efficiently. The problem this scenario depicts is referred to as the *Focus and Context Problem*.

The Focus and Context Problem is the difficulty the user faces in resolving where their current region of interest lies within the larger informa-

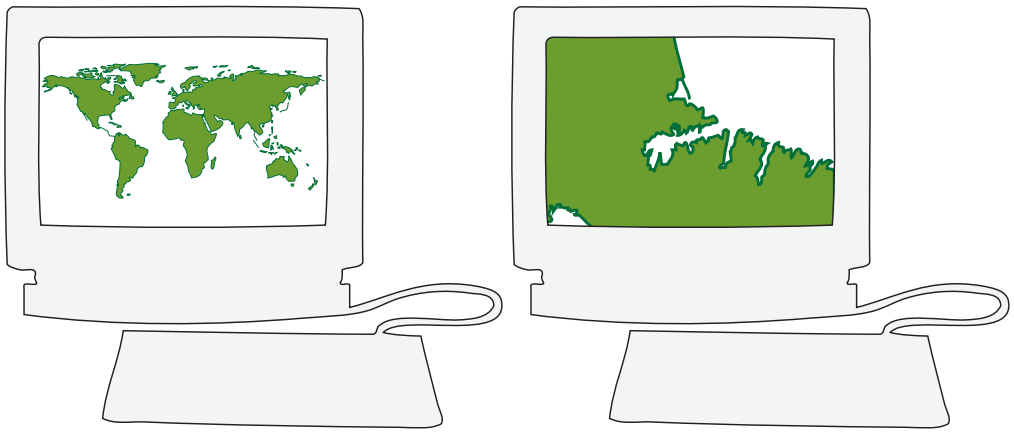


Figure 2.2: Loss of Focus and Context. The visualisation on the left provides an overview (Loss of Focus) and the visualisation on the right provides only a detailed view (Loss of Context). Reconciling the relationship between these two views is a challenge to the user that Focus and Context techniques aim to address.

tion space (Ware 2004). This can range between a complete *loss of focus* (an overview devoid of detail) to a complete *loss of context* (a detailed view that provides no clues as to where it belongs). These two situations are shown in Figure 2.2.

In this section the Focus and Context Problem has been introduced. A large information space and a small region of interest introduce conflicting requirements for global overview and local detail. Many visualisation techniques have been proposed that aim to provide the user with the necessary cues they need to maintain awareness within the user interface. These techniques are called Focus and Context techniques.

In the remainder of this chapter, the ways in which Focus and Context techniques work is examined. The first step involves considering the GUI as a collection of Views, as defined by the Model-View-Controller software design paradigm. By then examining the various ways that Views can be created, combined, and arranged together, it is possible to categorise all the different Focus and Context techniques into a small set of fundamental types. This decomposition forms the basis for a new taxonomy of Focus and Context

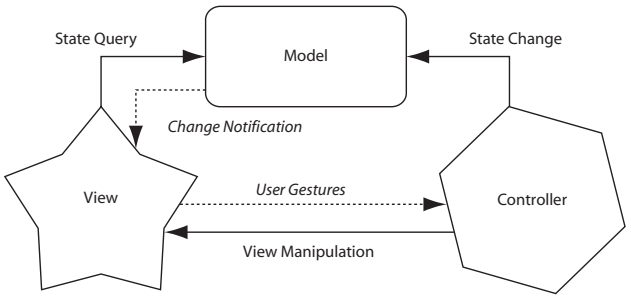


Figure 2.3: The Model-View-Controller (MVC) Architectural Pattern. MVC modularises the concerns of data, display and coordination.

techniques, which is then populated with examples from the literature.

2.1.2 MVC: The Separation of Views from Models

The GUI is the only part of a complete software system that most users see. A popular software development paradigm is the Model-View-Controller (MVC) (Burbeck 1992) architectural pattern.[1] In the MVC approach, the GUI is the *View*, which is combined with a *Model* (containing data and state information), and a *Controller* (to coordinate updates) to form a complete system. This configuration is shown in Figure 2.3. Of course, the modules need to communicate with each other. This communication is illustrated as the arrows in the figure and is summarised below:

- The View queries the Model for information to display. This can be enhanced by the Model notifying the View of relevant changes.

- The View passes user interaction events on to the Controller. Based on these events, the Controller will make the appropriate changes to the Model and View.

Ideally, any of the three modules can be changed with minimal impact on the other two. For example, changing the way data in the Model is stored

[1] The term *pattern* in this context refers to an accepted approach to solving a particular problem based on experience and distilled wisdom.

from a simple text file to a relational database should require little, if any, modification to the View. Likewise, the Model has no regard for how the information it stores will be presented, for that concern lies with the View.

The view transforms data from the model into a graphical representation. Therefore, GUI design can be thought of as both the art and science of generating views. Creating a usable view requires considerable care and there are many examples where interface designs fail (see "GUI Bloopers" by Johnson (2000)).

Card, MacKinlay and Shneiderman (1999) define information visualisation as "the use of computer-supported, interactive, visual representations of abstract data to amplify cognition". Therefore, in the terminology introduced previously, the transformation of data in the Model into information in a View is the crucial element in this domain.

Although not all software projects are developed with the MVC pattern explicit in the design process, it can be useful to conceptualise these components when examining the software's operation. In other words, even if a program does not follow the guidelines of MVC, employing this abstraction helps to explain and critique the user interface it provides. This conceptualisation can be recursively applied to many user interfaces because interface components are nested. For example, it might be convenient to consider the entirety of the computer operating system in terms of MVC, or only one particular application, a pane within that application, or only one particular scrollbar inside that pane.

2.1.3 Multiple View Systems

The same model can be the source for many views. Different types of views can be appropriate in different situations. The user's role will often determine the type of view that is suitable. For example, a project management system may provide a manager with a concise summary of the state of a project, whereas a software developer may be presented with a verbose report describing all the remaining bugs to be fixed. In a more general case, many applications allow the user to choose between "expert" and "novice" modes. The interface then adapts itself accordingly by hiding seldom-used

features from novices and streamlining complex tasks for experts.

The type of task often determines which type of view to use. Different types of views have strengths in different areas. For example, a spreadsheet view of a table of data facilitates data entry and manipulation, whereas a graph of the same data exposes trends and provides insight. Figure 2.4 illustrates this point.

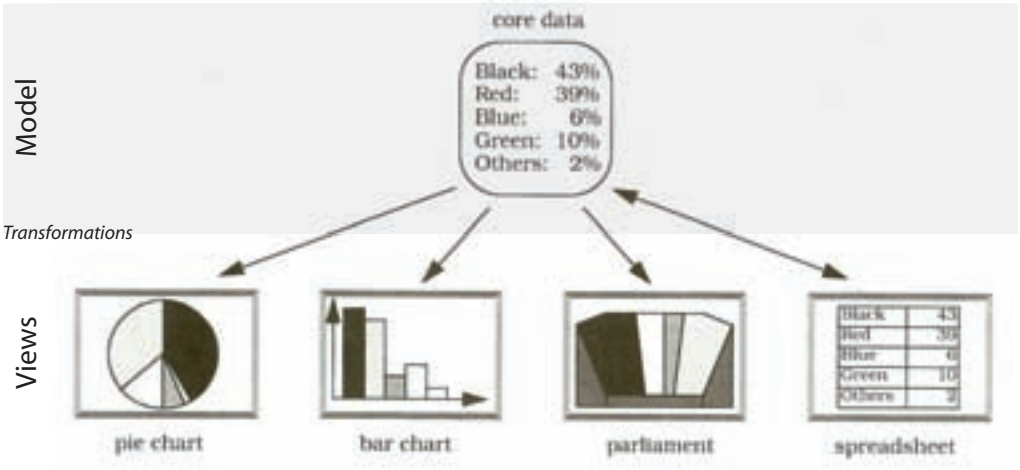


Figure 2.4: An example of the same data being represented in different ways. Note how the spreadsheet has a bidirectional relationship with the Model, indicating that it can be used to edit the Model. Diagram adapted from Buschmann et al. (1996), p.125.

Baldonado, Woodruff and Kuchinsky (2000) provide some useful definitions: a *single view* of a conceptual entity is the combination of a set of data and a specification of how to display that data visually. Two views are *distinct* if they differ in either the data they present, or the specification they use to present it. A *multiple view system* uses a combination of distinct views to support the investigation of a single conceptual entity. They give the example of Microsoft PowerPoint's interface, in which the main view displays the current slide in detail for editing tasks, and the thumbnail view at the side displays an overview of the presentation for ordering, insertion, deletion and browsing tasks.

So far we have considered the notion of *View* as the overall visual rep-

resentation of the software system. For example, the interface of Microsoft Word is a complicated view of a "document model". MVC can also be applied at an increasingly finer grain to all components that make up the interface - that is, views can be nested within each other. For example, the scrollbar used to navigate a document within Microsoft Word can be considered an independent view driven by the subset of the model concerned with the user's location within the document.

From a usability perspective, it is the user's current task that defines the logical separation of the user interface into views.

2.2 A Taxonomy of Views for Supporting Focus and Context

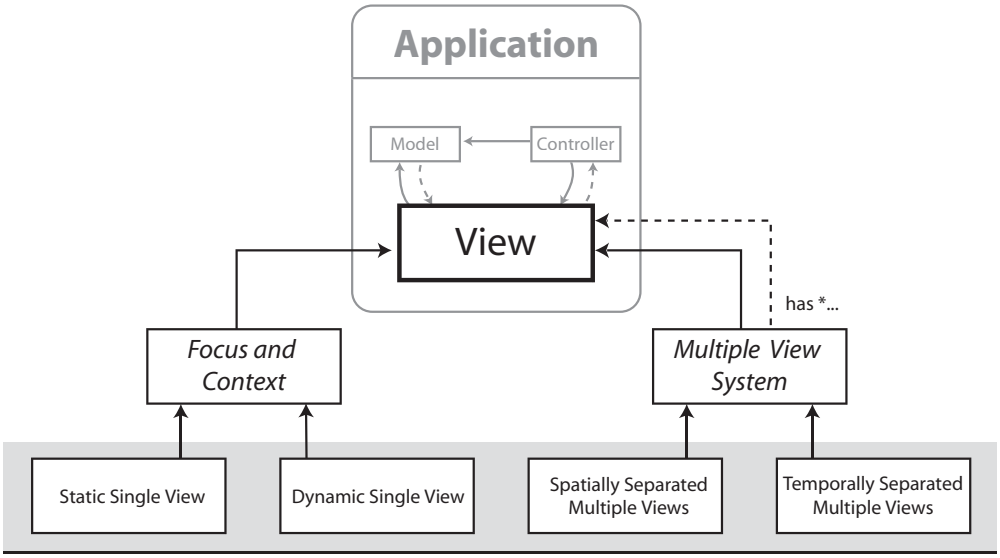


Figure 2.5: Taxonomy of Views for Supporting Focus and Context.

This section presents an organised classification of Focus and Context techniques. We have chosen an approach based on the concepts of Views and Models, where Views are the graphical representations of Models, generated through a transformation process. The hierarchy of our taxonomy is shown in Figure 2.5. This taxonomy provides the structure for the survey of Focus

and Context techniques presented in this section. As an introduction to the taxonomy, Figure 2.6 provides an example for each type of technique, using map browsing as a unifying scenario.

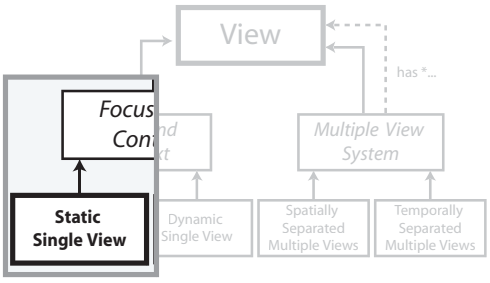
2.2.1 Single View Focus and Context

The Focus and Context Problem can be alleviated, to some extent, by carefully constructing views that present a mixture of detailed and contextual information. Björk, Holmquist and Redström (1999) defined a focus and context technique as a second-order visualisation, or "a visualisation of a visualisation". A visualisation can be thought of as a set of input data and the application of a display specification. For seamless single view focus and context techniques, this definition fits. For example, a fisheye lens view of a map can be thought of in two parts: the first visualisation generates the map itself from source data (a description of roads, towns and rivers). The second visualisation uses the result of the first (the map) as input and applies a spherical distortion to generate the final view.

Techniques that generate such views strike a balance between specific and general information so that the user does not become lost, confused or frustrated. This is achieved in one of two ways. Firstly, the view can continuously present both detailed and contextual information, or transition between the two over time. We will refer to the first case as Static and the second Dynamic.

Static Techniques for Single View Focus and Context

A static focus and context view amplifies details of interest and suppresses information elsewhere. As attention shifts to new items of interest the view adjusts accordingly. At any one point in time a static view will present both detailed and contextual information. These views often employ spatial distortion to bias the distribution of screen space in



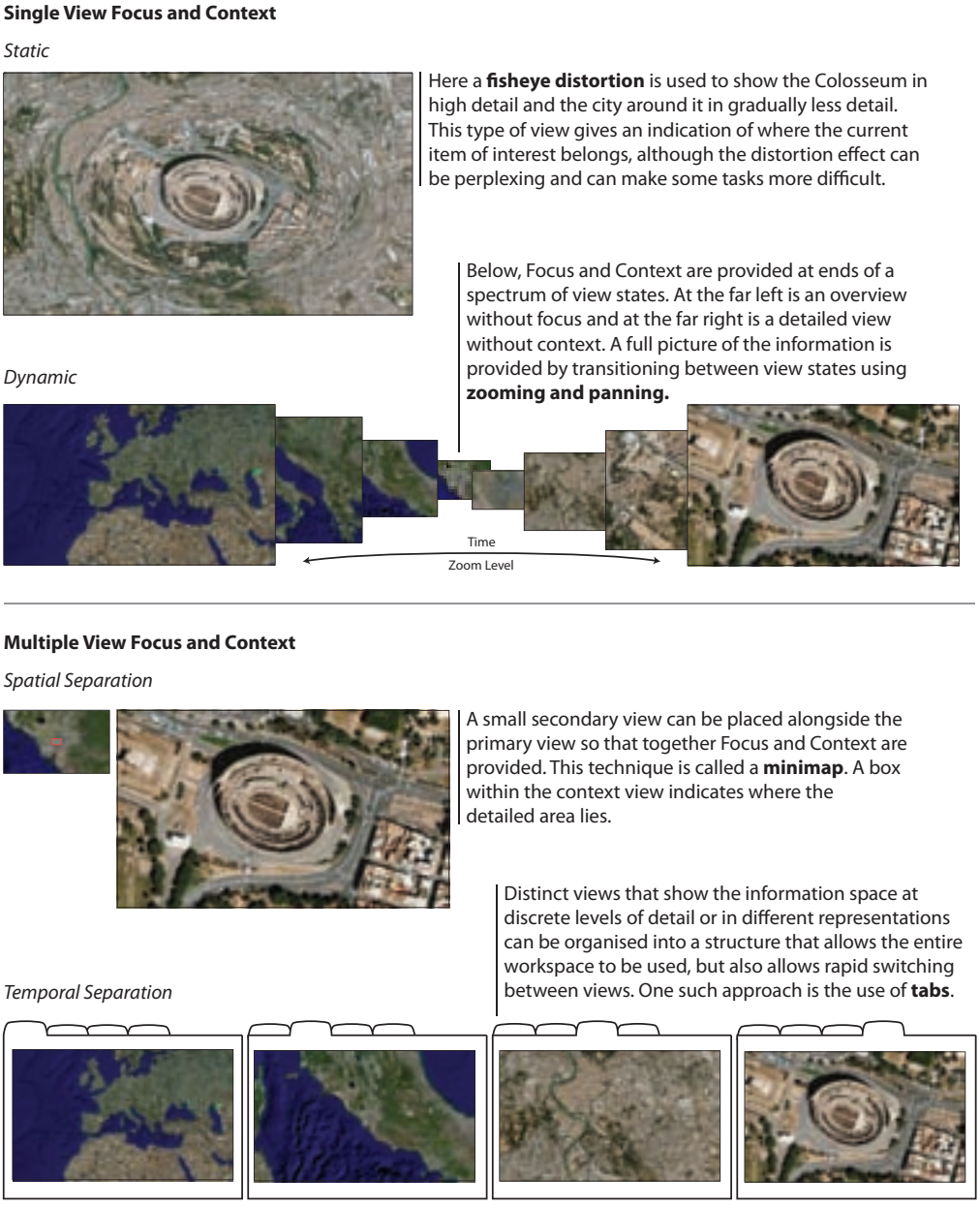


Figure 2.6: The above examples of Focus and Context techniques demonstrate instances of the four types of technique identified in the taxonomy. All the examples depict the visualisation of the Colosseum and its location within Rome, Italy, and Europe. Although a mapping scenario is used in these examples, it is important to note that the techniques can be applied in many different domains and some techniques are more applicable to certain types of data than others.

favour of interesting items.

One way of determining how to make this distribution is the notion of fisheye views (Furnas 1986). Fisheye views are a way to distill salient information from large and complex information structures based on a current item of interest. The degree of interest (DOI) of every other item is computed based on *a priori importance* and *distance*. An item's *a priori* importance boosts its DOI and its distance reduces its DOI (see Figure 2.7). Distance need not be Euclidean distance, but can be any measure of how far removed the item is from the item of interest. Provided with a DOI for all items, the view can adjust itself to accentuate items of high importance and de-emphasise items of low importance. As attention moves to a new item of interest the view updates to match the new array of DOIs.

$$
\begin{aligned}
DOI &= \text{degree of interest function} \\
API &= \textit{a priori} \text{ importance function} \\
D &= \text{distance function} \\
DOI_{fisheye}(x\ .\ = y) &= API(x) - D(x, y)
\end{aligned}
$$

Figure 2.7: The degree of interest (DOI) of an item x relative to y is the *a priori* importance of x minus the distance between x and y.

(a) A fish s view of the world. (b) A photograph taken with a fisheye lens. (c) The structure of a graphical fisheye.

Figure 2.8: The origin of the term "fisheye", a real world application in photography, and the geometrical layout of a graphical fisheye distortion.

The term "fisheye" originates from the distorted view of the world a

fish sees due to refraction of light at the surface of the water, as shown in Figure 2.8(a). It is also used to describe camera lenses with extremely wide fields of view that allow photographers to capture hemispherical images, such as the one in Figure 2.8(b). These photographs exhibit barrel distortion, which means that straight lines near the edges of the image appear to bend outwards, enlarging the centre of the image. The centre of the image is magnified, with detail diminishing toward the edges, as shown in Figure 2.8(c). This has become the widely accepted defining feature of graphical fisheye views.

However, Furnas (2006) emphasises that the fisheye DOI metric is independent of the visualisation technique used and is a means to select *what* information to display to the user, rather than *how* to display it. As such, a fisheye DOI approach can be used in non-visual applications, such as information retrieval systems and intelligent recommendation systems. Furnas recognised the fisheye arrangement of information in everyday situations. For example: an employee will have detailed knowledge of their own department within a large corporation but may only know the names of heads of other departments. A local newspaper will publish many banal local stories (low API, low D) and a few important global stories (high API, high D). People tend to know the names of influential world leaders (high API) and recent ones (low D), but forget the names of inconsequential ones (low API) and those long past (high D). These examples demonstrate that some items of information are generally important and others gain their importance from being closely related to the item of interest.

Furnas cited Saul Steinberg's 1976 humourous cover of the *New Yorker* magazine as an example of the fisheye view. The image depicts the world from the perspective of someone in New York (making that person the item of interest). Places within New York have low distance and high importance so they appear large and detailed. The rest of the United States is populated by only a few key landmarks in approximate locations, and the only labeled items beyond the Pacific Ocean are China, Japan and Russia.

There are also analogies of fisheye views in the physical universe. Gravity, for example, acts as an attractive force between all planetary bodies. The strength of the gravitational field around a body is proportional to that

body's mass (*a priori importance*) and inversely proportional to the square of the distance from the body's centre. For the planets in our solar system, we could choose one and work out the DOIs in terms of gravity for all the others. The inverse-square law also applies to light intensity and sound intensity emanating from a point source.

Graphical Fisheye Views were introduced by Sarkar and Brown (1992) as a way to apply the fisheye structures of Furnas to the domain of graph visualisation. The formalism introduced by Furnas, which described how to compute a *degree of interest* for all items, was expanded to include calculations for an item's new graphical position in the fisheye view, the size and detail it should be displayed at, and its *visual worth* (VW). VW was a measure derived from the item's undistorted distance from the focus and its *a priori importance*. VW was found to be a useful property of items. If an item's VW fell below a certain threshold, it could be culled from view. If two items overlapped, the one with the higher VW would be placed on top. Finally, in the early prototype used to demonstrate these views, response time was kept low by prioritising placement of items with high visual worth and approximating the positions for other items.

The Rubber Sheet Metaphor of Sarkar et al. (1993) again extended the concept of fisheye views to overcome two limitations of previous work. Firstly, in original graphical fisheye views, only a single item could act as the focus. Secondly, the amount of space given to the focus area was defined by the system, when it would be more appropriate for the user to control it. The metaphor of the rubber sheet was introduced as a solution to these problems. Following the metaphor, information is laid out on a rubber sheet and *handles* are used to apply stretching. Each handle defines the stretch between a source layout (when the sheet was grabbed) to a destination layout (where it is held in place). The rubber sheet with handles overcomes the limitations of previous graphical fisheye views by allowing the user to control which area is stretched (and therefore which items are stretched) as well as the strength of the stretch. For example, the map in Figure 2.9 shows the distorted view that results from enlarging the handles placed around

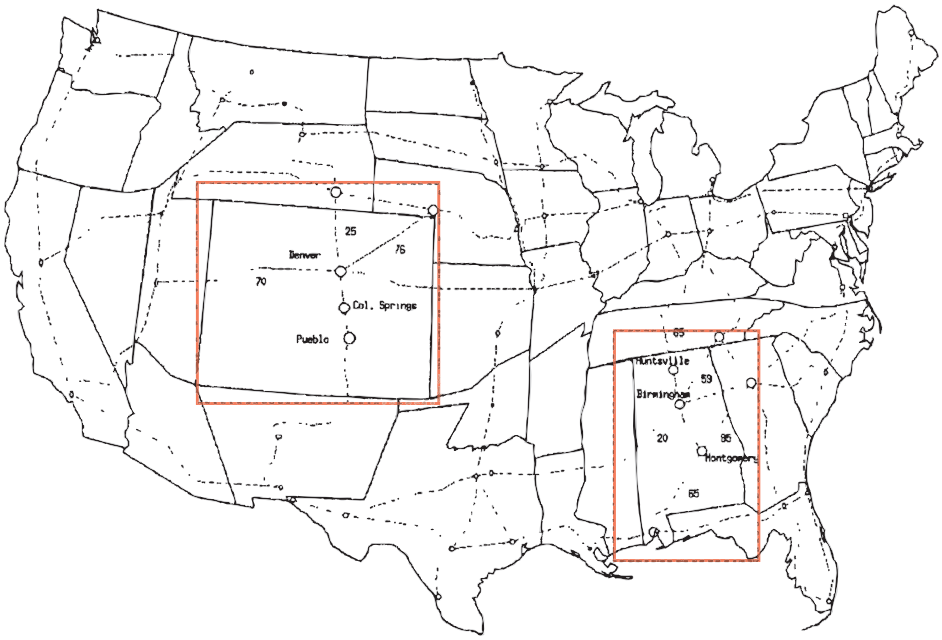


Figure 2.9: The Rubber Sheet Metaphor of Sarkar et al. (1993). The map is displayed as if on a pliable rubber sheet. Handles, highlighted in red, are used to stretch the sheet. In this example, the states of Colorado and Alabama have been stretched larger than normal size. Note that some labels within those states have appeared now that space permits.

the states of Colorado and Alabama. Leung and Apperley (1994) cited the rubber sheet metaphor as the unifying theory behind all distortion-oriented techniques.

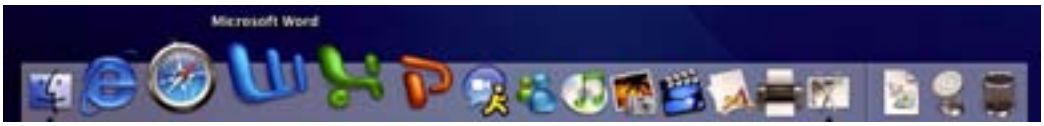


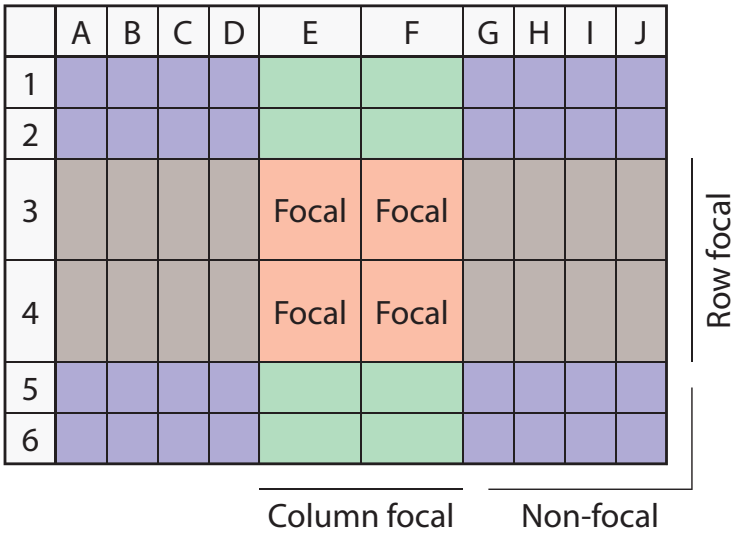
Figure 2.10: MacOSX Dock.

Fisheye views are now quite common in user interfaces. For example, the MacOSX dock uses a fisheye effect to enlarge the currently designated icon, with surrounding icons' size decreasing with distance (see Figure 2.10).

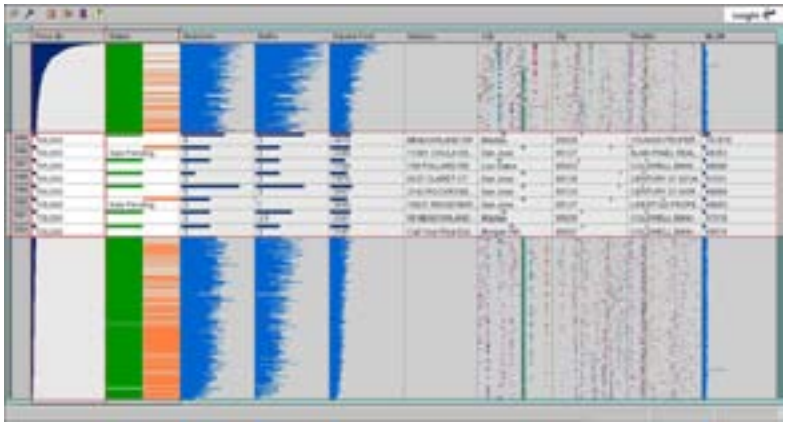
The Table Lens of Rao and Card (1994) applies a fisheye approach to visualising large tables of data. They claim the technique can display "up to 100 times as many cells" as a standard spreadsheet interface. This is achieved by compressing and expanding rows and columns to create areas of detail and areas of context. However, because rows are always perpendicular to columns, there are actually four types of area created: focal, row focal, column focal and non-focal, as shown in Figure 2.11(a). An example with real data is shown in Figure 2.11(b).

The Perspective Wall of Mackinlay et al. (1991), shown in Figure 2.12(a), presents information on virtual walls that diminish into the distance. It is particularly suited to linear data such as chronologically ordered information in a calendar. The Document Lens of Robertson and Mackinlay (1993) is a similar concept to the Perspective Wall, but extended to two dimensions to create a four-sided pyramid (see Figure 2.12(b)).

The Hyperbolic Tree of Lamping et al. (1995) is shown in Figure 2.13(a). They were motivated by the Dutch artist M. C. Escher (1889 - 1972) who created many clever tessellations in hyperbolic space, such as *Circle Limit IV*

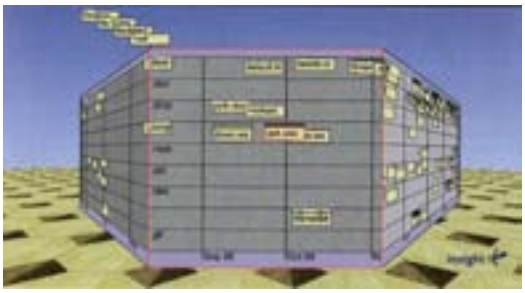


(a) Table Lens structure. The row height and column width for cells in the focus region are increased, resulting in four area types: focal (displayed large), row focal (enlarged rows), column focal (enlarged columns) and non-focal (original size).

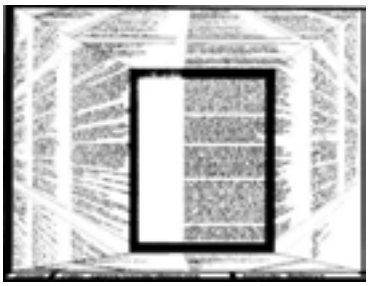


(b) An example of a Table Lens applied to real estate information. The data has been sorted by price (first column) and a range of properties have been selected.

Figure 2.11: The Table Lens of Rao and Card (1994) applies a fisheye approach to data arranged in a grid. Image courtesy of Ramana Rao.



(a) Perspective Wall of Mackinlay et al. (1991). Image courtesy of George Robertson and Stuart Card.

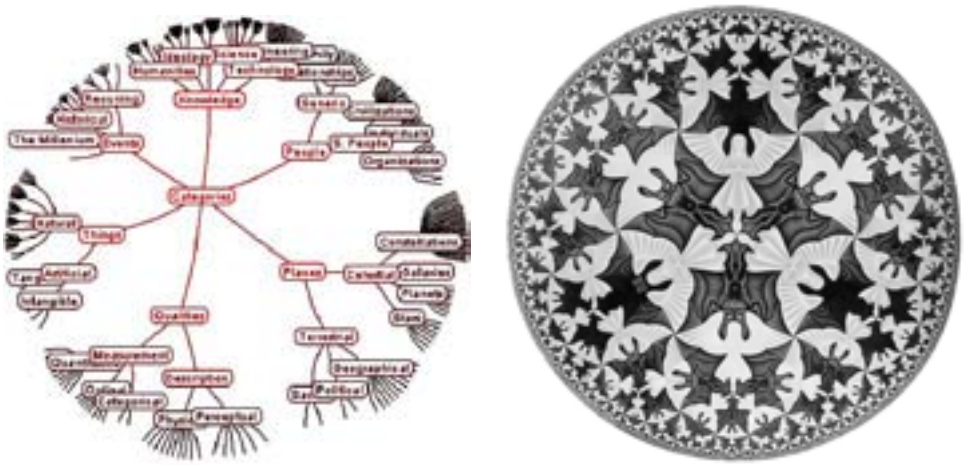


(b) Document Lens of Robertson and Mackinlay (1993). Image courtesy of George Robertson.

Figure 2.12: Static Techniques that employ distortion via perspective.

shown in Figure 2.13(b). In hyperbolic space the circumference of a circle grows exponentially with its radius. A mathematical model (such as the Poincaré model) is required to map hyperbolic geometry back to Euclidean space so that it can be viewed. This mapping necessarily causes distortion which is the property Lamping et al. (1995) exploit in their work. Unlike in Euclidean space, a tree hierarchy can be laid out in hyperbolic space with a uniform distance between all children and their parents, that is, all edges are the same length. When viewed in Euclidean space the sizes of nodes in the tree diminish the farther they are from the centre and the number of nodes increases exponentially from parent to child.

Cone Trees and Cam Trees are ways to visualise hierarchical structures in 3D (Robertson et al. 1991). The Cone tree, shown in Figure 2.14(a), is arranged vertically. A slight variation, the Cam tree, shown in Figure 2.14(b), is arranged horizontally, allowing for better display of text labels. When a node in the tree is selected, the entire hierarchy rotates to bring that node and its ancestors to the front. The illusion of depth created by the 3D perspective, lighting, and shadows, allows the display of many more nodes than would be possible in a 2D visualisation of the same dataset. The perspective view resembles a graphical fisheye effect, with the closest nodes appearing the largest.



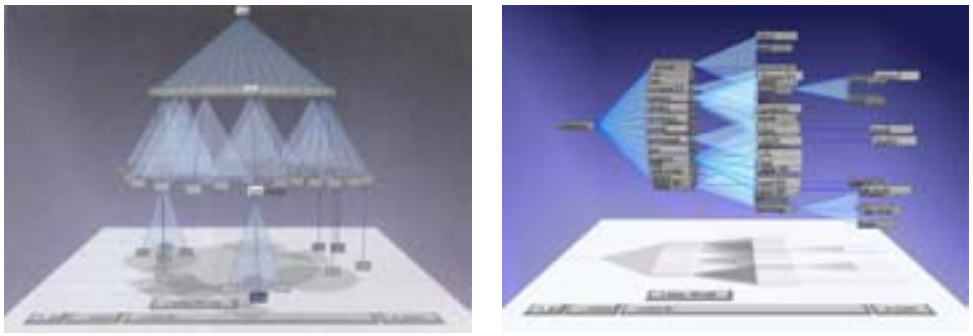
(a) Hyperbolic Tree of Lamping et al. (1995). Image courtesy of Ramana Rao.

(b) *Circle Limit IV* by M. C. Escher. All M. C. Escher works © 2008 The M.C. Escher Company - the Netherlands. All rights reserved. Used by permission. `www.mcescher.com`

Figure 2.13: Examples of visualisations of hyperbolic space.

Treemaps of Johnson and Shneiderman (1991) use a space-filling approach to visualise tree structures that are more frequently displayed as node-link diagrams. As shown in Figure 2.15, there is a one-to-one correspondence between the two types of diagram, the difference being that the vertically flowing parent-child relationships of the tree are replaced by nesting relationships in the treemap. Children are packed within their parent's region and assigned space based on their size or importance. The arrangement of child regions alternates between horizontal and vertical to produce obvious groupings.

Treemaps have been successfully applied in a multitude of domains. PhotoMesa is an image browser that uses the treemap algorithm to layout thumbnails of photograph collections based on directories, dates or keywords (Bederson 2001). Engdahl, Köksal and Marsden (2005) used treemaps to visualise threaded discussion forums on the small screen of a PDA and found its space-filling approach to be faster than text lists for finding the largest and most active threads. Balzer, Deussen and Lewerentz (2005) applied treemaps to



(a) Cone Tree. (b) Cam Tree.

Figure 2.14: Cone and Cam trees of Robertson et al. (1991). Images courtesy of George Robertson and Stuart Card.

the visualisation of software metrics, and also demonstrated an interesting treemap variation based on Voronoi diagrams, which produces treemaps with a organic, cellular appearance.

Semantic Depth of Field or SDOF is described by Kosara, Miksch and Hauser (2001) as a "cue" focus and context technique. SDOF selectively blurs less important items in the view so that they are literally "out of focus", as shown in Figure 2.16. In this chapter the term "region of interest" has been used to identify the part of the information space that is the current focus of attention. This term assumes that the user is interested in a spatial subset of the information, such as page 10 of a 125-page document, a 50 km radius around London, or the nine classes that directly inherit from `java.io.InputStream` in a UML class diagram. Although this has been the case with the Focus and Context techniques described thus far, sometimes users are more interested in aspects of the information space that cannot be defined spatially. For example, consider the task of identifying all railway bridges on a map of France. It is not possible to "zoom in" on all railway bridges because they are distributed across the entire map. It is scenarios like this where SDOF can be used to provide Focus and Context by blurring the map but keeping the railway bridges sharp. Therefore, in some cases it is more appropriate to think of a user's region of interest as a "subset

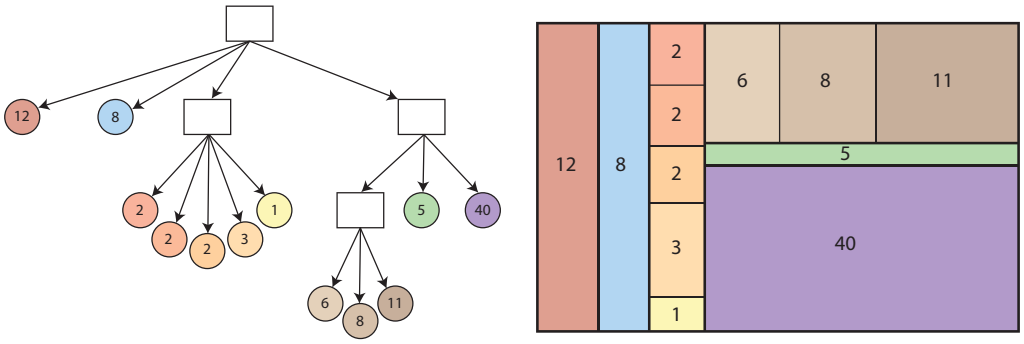
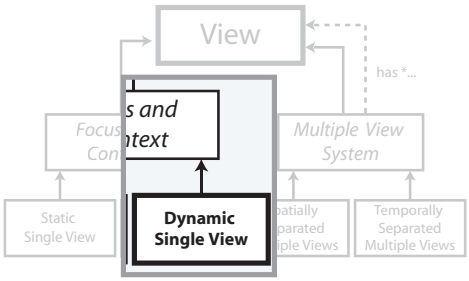


Figure 2.15: Example of a tree (left) represented as a Treemap (right). Child nodes hang beneath their parent node in the tree, but are packed within their parent's region in the treemap. Regions are divided based on the size or importance of child nodes. Diagram based on that of Shneiderman (1992).

of interest" because although the subset will be often be spatially bounded, other times it will be defined semantically by a query.

The Tag Cloud has become a common feature on websites, especially blogs, where content is categorised by topic keywords, or tags. The website can determine the popularity of different topics by counting how often pages with those tags are visited. The tags are then presented as a paragraph of links with the font size of each tag mapped to its popularity, as shown in Figure 2.17.

Dynamic Techniques for Single View Focus and Context



The second approach to single view focus and context uses a view that varies over time. As time passes, the interface transitions between detailed and contextual information, usually under the control of the user. Again, detail can be added to the view by optical enlargement, or increasing the level of semantic detail. This approach relies on the user's short

(a) 2D Chess view. (b) 3D Chess view.

Figure 2.16: Semantic Depth of Field selectively blurs objects of less relevance. In the chess example shown here, the visualisation is showing which white pieces are covering the knight at E3. Images courtesy of Robert Kosara.

Figure 2.17: A Tag Cloud. The size of each link in the cloud is determined by the popularity of the tag being linked to.

term memory to recall the previous state of the view, and to use this information to maintain awareness of where they are in the information space.

Temporal Focus and Context techniques use views that change over time to convey details as well as gradually build up an overview of the information space. It is the accumulation of knowledge through continuous viewing that provides the user with both focus and context.

Pan and Zoom is one way to assimilate large information spaces with limited screen space. With a fixed amount of screen space at a fixed resolution, we can choose to display a large amount of information at a low detail, or a small amount of information at high detail. Zooming is the process of setting this detail level. As we zoom in, we lose context but gain focus. As we zoom out, we lose focus but regain context. It is through repeated zooming that both focus and context are maintained.

When we zoom, we specify the subset of the information space we wish to view in higher detail. This selective process is controlled by panning. Panning allows us, at a certain zoom level, to control which subset of the information space is in view.

Together, panning and zooming provide the required functionality to visit any part of the information space at any detail level. However, these techniques alone have been found insufficient for many tasks, and many optimisations and variations of traditional panning and zooming have been proposed.

Many interfaces now do away with discrete zoom steps in favour of a technique where the user drags out a rectangle over the area of the workspace they wish to enlarge. This area is enlarged to become the new workspace. Often an action is available to jump back out to the previous zoom level.

Speed Dependent Automatic Zooming is a variation of Pan and Zoom that adjusts the user's zoom level based on their panning speed (Igarashi and Hinckley 2000). Panning speed and direction are controlled by a vector the user drags out with their mouse. When the user is panning quickly the interface decides they would benefit from more overview, so therefore zooms out. This is justified because the faster the user pans, the more forward knowledge they will require. When the user stops panning, the interface

Figure 2.18: Speed Dependent Automatic Zooming. With a single input mode (dragging the mouse) the user can navigate the entire globe. Images courtesy of Andy Cockburn.

assumes they have found what they were looking for and zooms back in to provide detail. Therefore, one input, rather than two, is required by the user to navigate the information space. This technique is illustrated in the globe browser of Cockburn, Looser and Savage (2003), shown in Figure 2.18. There are also several variations on the technique, such as speed-coupled flying (Tan, Robertson and Czerwinski 2000).

Semantic Zooming uses the current zoom level as a parameter to determine the representation of an object (Perlin and Fox 1993). This change in representation can be in addition to geometric zooming, or it can be an alternative. Semantic zooming is used commonly with maps. For example, as the user zooms in, the names of large towns appear first, and the names of small towns gradually appear as the zoom level increases. The idea is that detailed information about an area, such as the names of all its small towns, becomes increasingly relevant as you zoom in on that area. In the meantime, the interface may reduce overall clutter by hiding them.

Dynamic Queries are a technique that allows users to specify database queries using direct manipulation rather than through traditional computer languages like SQL, which requires time to learn and pre-existing knowledge of the database structure. Dynamic queries apply direct manipulation techniques to the problem of constructing database queries. That is, "rapid, incremental and reversible changes" can be applied directly to query parameters using sliders, for example (Williamson and Shneiderman 1992). The

query is tightly coupled with the view so that any change to the parameters becomes immediately evident in the display. This property encourages exploration and increases the user's confidence. Examples of a language-based query interface and a dynamic query interface are shown in Figure 2.19.

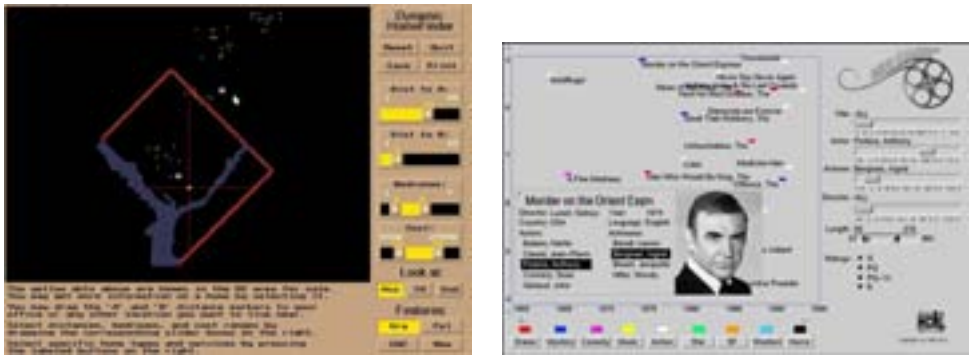
(a) SQL Interface.

(b) Dynamic Query Interface.

Figure 2.19: SQL versus Dynamic Queries. Both interfaces can be used to access sufficient information to complete the task of finding a house that meets the correct criteria. The SQL interface may be more powerful, but it also requires a much higher understanding of both the syntax of the query language and the structure of the database. Diagrams inspired by the HomeFinder of Williamson and Shneiderman (1992).

An early example of a system that employed this technique was the HomeFinder of Williamson and Shneiderman (1992). This interface, shown in Figure 2.20(a), provided an easy way for users to quickly filter a set of approximately 1000 homes using dynamic queries. Sliders were used to establish the ranges for values in the query, such as the number of bedrooms, the distance from key locations, and so on. The interface was found to be significantly faster than a natural language querying system (in which the system responded to questions typed in English sentences) and traditional paper printouts of the database contents (three versions were provided, sorted in

various ways).



(a) HomeFinder uses dynamic queries to control the acceptable ranges of house price, distance from workplace, and number of bedrooms. Houses not matching the criteria are hidden from view. (Williamson and Shneiderman 1992)

(b) FilmFinder presents films on a 2D scatter-plot, where the y-axis rates the popularity of the film and the x-axis shows the year. The desired ranges can be set on each axis to reduce the number of visible items. (Ahlberg and Shneiderman 1994b)

Figure 2.20: Interfaces that use dynamic queries to filter information. Used with Permission University of Maryland Human-Computer Interaction Lab, `http://www.cs.umd.edu/hcil`.

Williamson and Shneiderman (1992) describe many advantages of dynamic queries. They are quick to set and provide rapid feedback and tuning. They are easy to learn and prevent errors by not permitting illegal values or incorrect syntax to be entered. In contrast, there are also drawbacks to using dynamic queries. Although not such an issue with faster computers, the real-time updates required by a dynamic query interface can raise performance concerns. Variables in the queries must be able to be ordered and fit a range so that they can be input via sliders (or other widgets). A further disadvantage is that a custom interface may be required for each new dynamic query application, whereas database query languages are completely general.

Some of these issues have been addressed by later work, such as the StarField display of Ahlberg and Shneiderman (1994a) which generalises the dynamic query approach so that it can be applied to any domain, such as films in FilmFinder (Ahlberg and Shneiderman 1994b), shown in Figure 2.20(b).

Fishkin and Stone (1995) applied Magic Lenses (which will be explained later) to the specification and construction of dynamic queries. A Magic Lens is a bounded sub-region of the workspace which applies a transformation to the data objects within its border. In terms of the Focus and Context taxonomy, a Magic Lens is a spatially-separated multiple view technique, rather than a dynamic single view technique. However, as the interface of Fishkin and Stone (1995) relates to dynamic queries, the work is presented here.

One of the drawbacks of the original dynamic queries of Williamson and Shneiderman (1992) was that it was difficult to offer complex Boolean queries, such as union ("a house with four bedrooms and one bathroom OR a house with two bedrooms and two bathrooms") or negation ("NOT a house with two bedrooms"). Magic Lenses were used as a solution to this problem, while maintaining the direct manipulation feel of dynamic queries.

Multiple lenses can be active simultaneously, and by associating a dynamic query to each one, complex queries can be specified via the intersection of overlapping lenses. An operator assigned to each lens determines how queries should be composed. Queries and operators are input using sliders and buttons attached to the lenses. The mechanism is demonstrated in Figure 2.21, in which the two lenses overlap to select U.S. cities with high salaries OR low taxes.

Rapid Serial Visual Presentation or RSVP allows a large amount of information to be digested in a short space of time (de Bruijn and Spence 2000). This is achieved by presenting chunks of information, such as pages, words or pictures, one after the other in quick succession. This technique is described by the authors as the "electronic equivalent of riffling a book in order to assess its content".

There are several modes in which RSVP can be presented. The most basic is Keyhole, in which each item is displayed in succession in the same space. In Carousel RSVP, image thumbnails rotate around the screen, increasing in size to a maximum at the 12 o'clock position and then shrinking again. Collage RSVP builds up a display by dropping new image thumbnails on top of old ones. Floating RSVP appears as though the user is moving forward

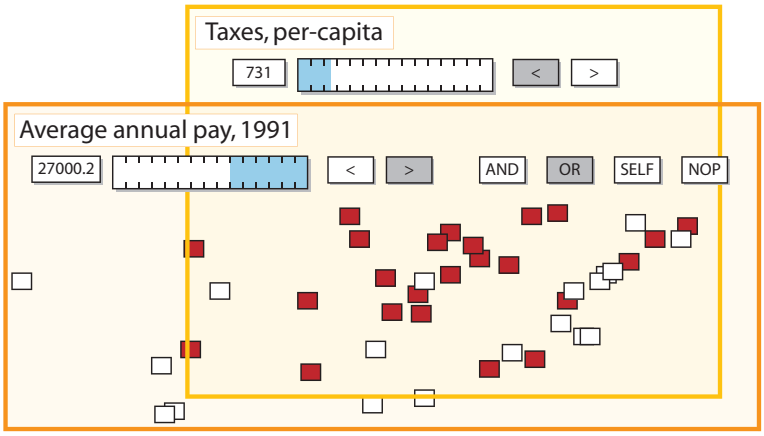


Figure 2.21: Magic Lenses for Dynamic Queries. The intersection of the two filtering lenses highlights cities with high salaries OR low taxes. Diagram reproduced from Fishkin and Stone (1995).

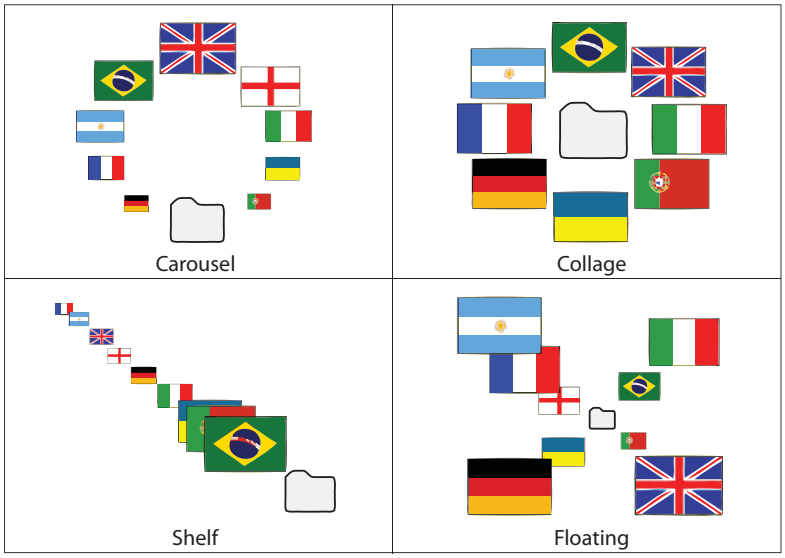


Figure 2.22: Four different modes of Rapid Serial Visual Presentation (RSVP). RSVP presents a large amount of information by displaying information items in quick succession. Diagram based on that of de Bruijn and Spence (2002).

through space, passing the images like one passes billboards on a highway. Finally, in Shelf RSVP, image thumbnails move across the screen, growing larger and then smaller again. Carousel, Collage, Floating and Shelf RSVP modes are illustrated in Figure 2.22.

2.2.2 Multiple View Focus and Context

The previous section described views that provide both Focus and Context either simultaneously (static) or through changes over time (dynamic). In this section, we describe views that work together to provide Focus and Context. On their own, many views cannot be classed as "focus" or "context" because such a classification can only be defined in terms of a particular task. For example, a map of Europe is neither an overview or a detail view. If the user is presented with a task like "Find all universities within three hours of Paris", the map gives context. However, for the task "Locate all cities in the world with populations over one million" the map is clearly providing focus for but part of the overall task.

Panes are the building blocks of the 2D graphical user interface. They are sometimes referred to as panels, frames or containers, depending on the tools used to design and implement the interface. Panes provide the organisational structure that defines the arrangement of multiple views in most 2D user interfaces. Figure 2.23 shows the layout of panes in Microsoft PowerPoint and Autodesk 3D Studio Max.

In the taxonomy presented in this chapter, a particular type of view is the multiple view system. This type of view displays a set of child views, rather than directly visualising information. These child views may themselves be multiple view systems, leading to the ability to construct complex hierarchical interface structures that arrange UI components. Panes are the mechanism by which most windowing toolkits provide this ability. For example, a simple chat program may have two views: a large text area for all chat messages, and a single line text box for the user to type their current message. These two views can be packed within a pane, which becomes the root (or main window) of the chat program.

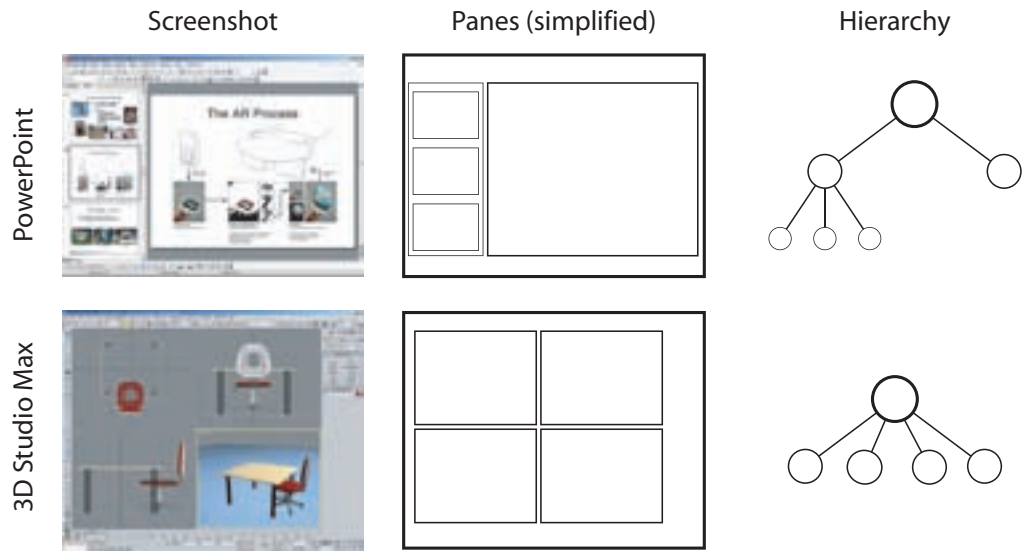
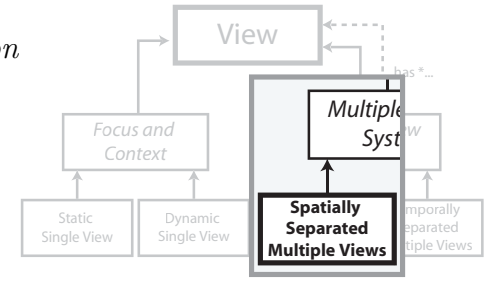


Figure 2.23: The layout of panes in two applications. The left column shows a screenshot of the application. The middle column shows the layout of panes within the application. The right column shows a tree representing the relationships between panes.

There are two basic ways to arrange panes. Firstly, the parent pane can be partitioned to display more than one view at the same time. Here, space is traded-off for continuous display. Alternatively, panes can occupy the same space at different times. These two possibilities are referred to as spatial and temporal separations respectively. By creating a multiple view system consisting of both focus and context views, the combination can be considered a Focus and Context technique. When multiple views are employed in this way, interface designers must decide the best way to arrange them in the finite screen space available. Techniques that employ each of these approaches are described in detail in this section.

Multiple Views with a Spatial Separation



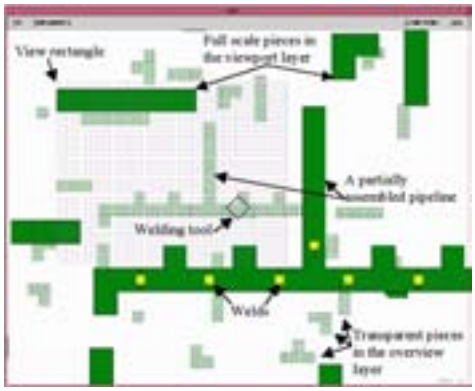
Layers can be considered a spatial separation in the z-axis (depth), where semi-transparent panes can be

overlaid to build up information-rich interfaces. This technique is used by many people to handle complexity in everyday tasks. For example, architects layer traced building plans to show complete structures from simpler component drawings. Teachers combine acetate overhead transparencies to break down complicated concepts for their students. The technique is also commonly used in user interface prototyping, where mock up interface components and user input can be quickly simulated. There are many other applications for this approach.

Alpha-Blending is a compositing technique that computes the final colour of a pixel as the weighted sum of the colours from the foreground and background layers. When applied to user interface design, additional display space can be created by overlaying multiple full-screen panes and then alpha-blending them. The ratio (alpha) that determines how much each layer colour contributes to the final colour can typically be defined by the user.

Using alpha-blending in an Overview+Detail interface allows the overview to be as large as the workspace, rather than a separate window which must share screen space with the detail view. Cox et al. (1998) showed that layering a semi-transparent overview on top of a detailed view can be a usable solution. In a task where scattered graphical objects had to be arranged to match a target configuration, users were observed using both layers easily, sometimes even simultaneously, to complete the task (see Figure 2.24(a)). Higher levels of transparency (i.e. more see-through) were preferred and confusion between layers began to emerge as objects in the overview became more opaque (i.e more similar to the detail view).

Multiblending by Baudisch and Gutwin (2004) is an enhancement of alpha-blending. It uses image processing techniques to blend features of overlapping windows more intelligently than simply taking a percentage of each layer's colour. Alpha-blending can reduce the readability of views due to the ways in which colours and textures combine. Multiblending takes an approach grounded in perception, where the critical features of the foreground layer are maintained and as much of the background layer as possible is left unobscured. In tests of how well foreground and background layers



(a) Alpha-Blending. The workspace has semi-transparent elements on an alpha-blended layer. Image from Cox et al. (1998).

(b) Multi-Blending enhances alpha-blending by intelligently combining layers. Image from Baudisch and Gutwin (2004).

(c) Free Space Transparency only blends the unimportant regions of the workspace. Image from Ishak and Feiner (2004).

Figure 2.24: Layers with various blending techniques. Images (a) and (b) courtesy of Carl Gutwin, (c) courtesy of Steve Feiner.

(a) The City Lights technique uses window borders to display cues about off-screen objects. Image courtesy of Mark Stefik.

(b) The Halo technique uses the screen border to show the edges of circles centred on objects that lie outside the screen area. Image courtesy of Patrick Baudisch.

Figure 2.25: Visualisation techniques that provide context from peripheral cues.

were recognised, a glass-like Multiblending effect, shown in Figure 2.24(b), was found to be at least as fast as alpha-blending (at various transparency levels), significantly faster for some types of images, and generally preferred over alpha-blending.

Free Space Transparency or FST is a further refinement of alpha-blended layers (Ishak and Feiner 2004). FST attempts to remove the ambiguities introduced by transparency by ensuring that only unimportant regions of the user's workspace are blended. Important regions remain opaque and gradients are used to smooth the transition between regions (see Figure 2.24(c)). A large problem for this technique is determining which window regions are important and which can be blended. One solution is for individual applications to calculate and report these regions, but this is unlikely to ever be widely supported. Alternatively, an external method such as an eye-tracker could potentially be used to record which regions the user's eyes dwell on least often.

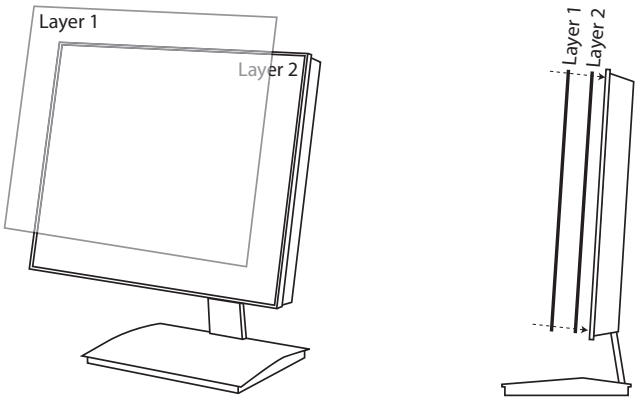


Figure 2.26: The PureDepth Multi-Layer Display is a physical display that spatially separates views.

City Lights by Zellweger, Mackinlay, Good, Stefik and Baudisch (2003) is a visualisation technique that provides information about off-screen objects (objects that are not currently within the bounds of the viewport). The thin borders of windows and sub-windows are used as a display space, within which a variety of cues about off-screen objects can be visualised. For example, simple points can indicate the existence of an off-screen object, or a line can indicate the size of the object, as shown in Figure 2.25(a). Colour can show additional information, such as the distance of the object.

Halo is a variation of City Lights designed for small-screen devices (Baudisch and Rosenholtz 2003). Circles centred on relevant off-screen objects are drawn large enough so that they intrude slightly into the display space. The curvature of the visible arc immediately conveys the approximate distance and direction to the target. For example, a wide, flat arc must necessarily have a large radius and therefore the object is far away. A close object will have a rounder arc. The direction is obvious from the position on the screen edge the arc appears. These relationships are illustrated in Figure 2.25(b). Compared to an interface using arrows to point to off-screen targets, Halo was up to 33% faster for various navigation tasks, and was the preferred technique.

A physical separation between layered views is also possible. The PureDepth multi-layer display (MLD) overlays two physical LCD panels so that information can be presented at two discrete depths, as shown in Figure 2.26. The front panel is transparent wherever the colour white is drawn, exposing whatever is being displayed on the back layer. The layers are separated by 7 mm although this distance appears to be as large as 14 mm on some models. This type of display is a relatively new technological development and as yet there are few studies to evaluate its effectiveness.

Masoodian, McKoy, Rogers and Ware (2004) modified a word processor to show a detailed document view on one layer of the MLD and a zoomed-out thumbnail view on the other layer. This system, called DeepDocument, allowed the user to access both detailed and contextual information simultaneously. Unfortunately, there seems to be no formal evaluation of this system.

Wong, Joyekurun, Mansour, Amaldi, Nees and Villanueva (2005) investigated the MLD and suggested a set of properties of the device that may make it superior to single layer displays for information visualisation and management. They proposed that the MLD may suit Focus and Context applications, where items of interest can be brought to the front layer to focus attention. The MLD may also have advantages for dense data, where partial occlusion of data points can be alleviated through depth and parallax (as the user moves their head, the overlapping points will appear to move at different rates).

Another example of a physical separation is the focus+context screen of Baudisch, Good and Stewart (2001), which uses a large projection screen to present contextual information while a smaller embedded high-resolution screen provides details (see Figure 2.27).

Thumbnails are scaled-down versions of other views. The reduction in size means that multiple thumbnails can be displayed in the same space as a single detailed view. If the information space can be partitioned into logical chunks, each chunk can be represented as a thumbnail to provide an overview of the entire space. Common examples of this situation include pages in a document (Figure 2.28(a)), slides in a presentation (Figure 2.28(b)), or

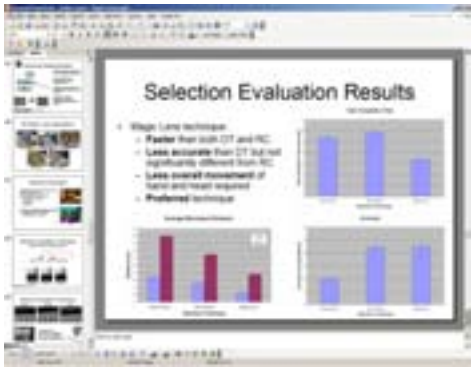
Figure 2.27: A Focus+Context screen embeds a small high-resolution LCD screen within a large but low-resolution projected screen. Image courtesy of Patrick Baudisch.

currently running applications (Figure 2.28(c)). Thumbnails are often used to create a visual index of a collection of items. This approach has become ubiquitous in online shopping catalogues, picture galleries and desktop file browsers.

Worldlets are an interesting extension of thumbnails for use in 3D virtual environments (Elvins et al. 1998). When a worldlet is created, the user's current view of the 3D world is copied, and clipped to a reasonable view volume. Worldlets in a gallery can be manipulated and explored in thumbnail form, as shown in Figure 2.29, and then activated to return the user to the location at which it was captured. Elvins et al. (1998) evaluated worldlets against landmark descriptions provided as text strings or images. Having 3D interactive thumbnails greatly enhanced participants' ability to recall locations and, when tasked to return to those locations, they required significantly less time and travelled a shorter overall distance with virtually no backtracking, compared to normal thumbnails or text labels.

Scrollbars are a compact way of providing contextual information about a neighbouring view, as well as facilitating control over the region of interest displayed in that view. A properly designed scrollbar can tell the user, at a

(a) Acrobat Reader uses a thumbnail view to provide an overview of the document and quick access to pages.



(b) Powerpoint uses a thumbnail view to provide overview, quick access to slides and also a way to easily arrange and reorder slides.

(c) Windows Flip is a new task switching interface in Windows Vista. Rather than displaying the icon representing a running application, a live thumbnail view of that application s window is displayed instead.

Figure 2.28: Thumbnails.

Figure 2.29: Worldlets are 3D interactive thumbnails that can be used as bookmarks into a virtual environment. This image, taken from Elvins et al. (1998), shows a gallery of worldlets. The selected worldlet (bottom right) can be viewed from any angle and distance using the controls around it. Clicking "Goto" enters the virtual environment at the location pointed at by the selected worldlet. Image courtesy of David Kirsh.

glance, where they are in a document, and roughly how large the document is. Scrollbars are one dimensional, meaning that one scrollbar is required for navigating each dimension of an information space. This makes scrollbars suitable for linear documents and lists, but they can become unwieldy for large images and maps where 2D navigation is desired.

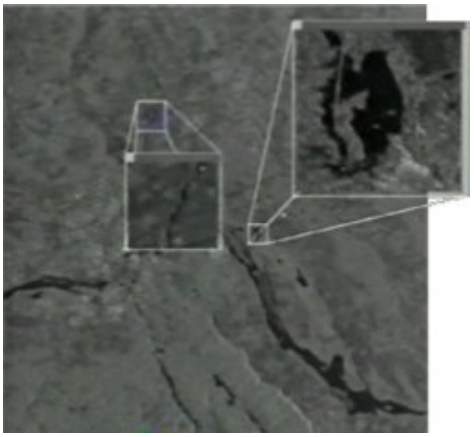
Multiple windows allow the user to position views at will. They can be laid out spatially in a tiled arrangement, layered and accessed one at a time, or positioned at random and accessed as required.

Lenses can be considered a hybrid multiple view technique because they combine aspects of both spatial and temporal separations. The lens provides a secondary view, embedded within the primary view, which can be moved around to display details about different regions at different times. The lens view modifies the presentation specification of the primary view to provide a magnified representation.

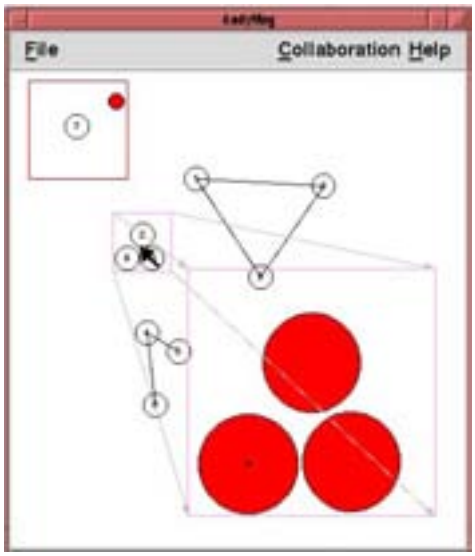
Magnification tools have a long history in document and image manipulation tools, such as `xdvi` shown in Figure 2.30(a). Such tools aim to provide the user with a magnified view of the region of interest, while maintaining surrounding context. One disadvantage of this approach is that the region of interest is covered by the magnified view. Several variations have attempted to address this problem, such as the DragMag interface of Ware and Lewis (1995), shown in Figure 2.30(b), and the Offset Lens of Greenberg, Gutwin and Cockburn (1996), shown in Figure 2.30(c). These interfaces introduce an offset between the region of interest and the magnified version, using lines to indicate the relationship.

An additional benefit of magnification lenses is that although the view is enlarged, the motor-space remains constant, allowing for precise mouse actions within the lens area. This characteristic was the basis for the Pointing Lens interface of Ramos, Cockburn, Balakrishnan and Beaudouin-Lafon (2007), shown in Figure 2.30(d), in which precise stylus input on a PDA was made possible by temporary magnification lenses that could be instantiated by methods such as stylus pressure.

(a) xdvi. A program to display TEX.dvi files.



(b) DragMag. Image courtesy of Colin Ware.

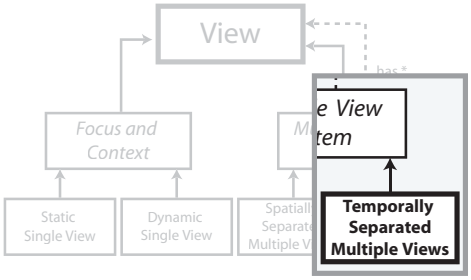


(c) Offset Lens. Image courtesy of Andy Cockburn.

(d) Pointing Lenses. Image courtesy of Andy Cockburn.

Figure 2.30: Magnification lenses display an enlarged version of the user's region of interest. Variations introduce an offset between the input and output regions, connected with lines to indicate the relationship.

Multiple Views with a Temporal Separation



Tabs are a method of arranging panes in a user interface where each pane occupies the same space. An additional control, in the form of a row of stylised buttons, provides access to each pane. When a button is clicked, the associated pane becomes the one and only pane visible. Tabs are completely ubiquitous in current operating systems and applications, such as Mozilla Firefox[2], which introduced the much-celebrated tabbed-browsing method of web navigation.

Tabs themselves do not provide Focus and Context. Like panes, tabs are merely a method of arranging different views. The content of the views themselves determine the usefulness of the visualisation.

Space Filling Thumbnails or SFT are a variation on the traditional use of thumbnails that aims to remove the need for scrolling in large documents (Cockburn, Gutwin and Alexander 2006). Thumbnails normally accompany a detail view in a spatial arrangement (described earlier), with a column of thumbnails at the side for navigation and other tasks. In contrast, SFT uses a temporal separation between the detail and thumbnail overview. In the overview (shown in Figure 2.31), a thumbnail grid of all pages in the document is displayed. Scrolling is not required because the grid is the size of the workspace, and the pages always stay in the same position, exploiting spatial memory.

2.2.3 Summary of Focus and Context

Simple approaches for visualising large amounts of information can become unwieldy and insufficient. A common problem is maintaining awareness of where one's current "region of interest" lies within the data, or, maintaining "focus and context". There are a wide range of methods for providing focus

[2] Mozilla Firefox, `http://www.mozilla.org/`, online as of 28 September 2007

Figure 2.31: Space Filling Thumbnails uses a contextual view, shown in this diagram, to display a grid of thumbnails of all pages in a document. The user can quickly select a page to visit, at which point the thumbnails disappear and the detailed view returns.

and context. In this section, the notion of a View (as defined within the Model-View-Controller paradigm from software engineering) was used as a way to distinguish between different focus and context techniques. The View is the module of a software system responsible for presenting information (stored in the Model) to the user. Four basic approaches were identified, a single static view, a single dynamic view, spatially separated multiple views, and temporally separated multiple views. This classification provided the structure for a large survey of focus and context techniques.

One particular technique, the Magic Lens, shows particular promise as a tool for managing information. The Magic Lens differs from standard desktop magnification tools by using a generalisable transform from the Model to the View. This permits an unlimited range of effects, visualisation techniques and interaction styles. In the next section we describe Magic Lenses in more detail.

2.3 Magic Lenses

A Magic Lens is a semi-transparent user interface element that allows operations other than magnification to be applied to the underlying content (Bier, Stone, Pier, Buxton and DeRose 1993). Whereas a standard magnifier could enlarge the view to expose more detail, a Magic Lens could, for example, enlarge the view, introduce useful labels and highlight items of particular interest.

A Magic Lens is an example of a direct manipulation interface object, as defined by Shneiderman (1987). That is, it provides a continuous representation of objects of interest and allows rapid, reversible, incremental actions and feedback. Users directly manipulate objects presented to them, using actions that correspond, at least loosely, to the real world. Users are familiar with magnifying glasses from real-world experience and should therefore understand the idea of looking through a lens and seeing something in a new way.

The Magic Lens concept was introduced by Bier et al. (1993) at Xerox PARC and was part of the revolutionary ToolGlass interface (see Figure 2.32). Among other things, Toolglass demonstrated the use of bimanual in-

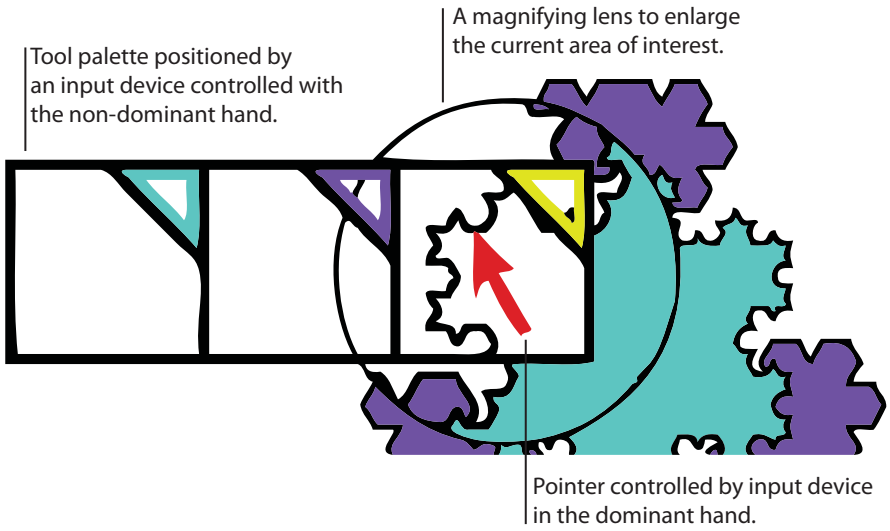


Figure 2.32: The ToolGlass interface supported semi-transparent tool palettes that were positioned over the workspace with one hand and "clicked-through" by the pointer controlled by the other hand. Diagram adapted from Bier et al. (1993).

teraction, designed in accordance with kinematic chain theory (Guiard 1987). That is, the user's non-dominant hand is good for rough placement and provides a frame of reference for precise actions made by the dominant hand. Toolglass exploited this capability by placing a semi-transparent palette of tools under the control of the non-dominant hand, and the cursor under the control of the dominant hand. To apply an operation to an application object, the user simply moved the palette until the appropriate widget was over the desired application object, and then clicked through the palette with the cursor.

To enhance the visualisation and interaction capabilities of ToolGlass, special Magic Lens filters could be incorporated into widgets, such that as the palette of widgets moved over the workspace, the representation of the region covered by the widgets would change. This allowed the workspace to be tailored for each particular tool. For example, when selecting a vertex of a shape, the selection widget could incorporate a Magic Lens filter that highlighted vertices and displayed vertices that would have otherwise been

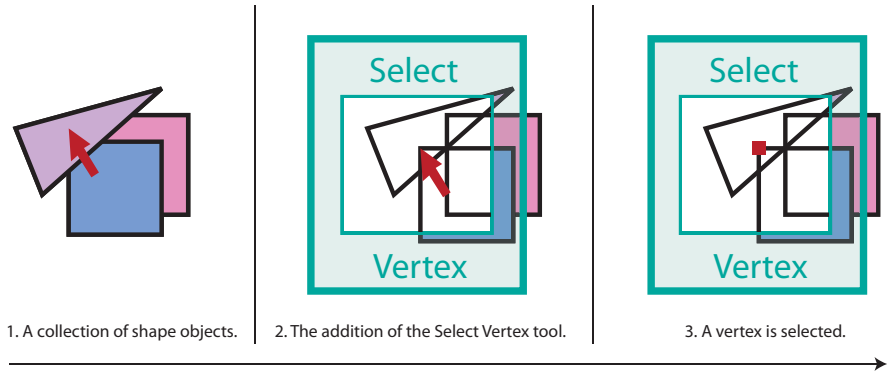


Figure 2.33: A tool to ease the selection of vertices in a graphics application. As the tool, with embedded Magic Lens, is brought over the shapes, the Magic Lens alters the visualisation to wireframe so that previously hidden edges are visible. The interaction style is adjusted so that the mouse pointer snaps to vertices. Diagram adapted from Bier et al. (1993).

hidden behind other objects (see Figure 2.33).

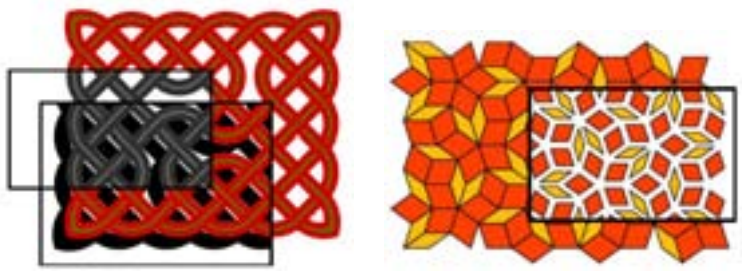
Magic Lenses could also be used as individual interface components. A lens could be placed within the workspace and be dragged about using the mouse. This type of interaction is useful for investigating areas of interest. Simply dragging the lens into the area could uncover details or a secondary layer of information.

Multiple lenses could be instantiated and active at once and the intersection of multiple lenses could present a composite transformation. Techniques to implement this behaviour are described in Section 3.1.

2.3.1 2D Magic Lens Systems

In this section, previous Magic Lens systems are presented. Although originally designed for 2D workspaces, the Magic Lens concept has been applied in 3D desktop and immersive virtual reality applications. Traditional 2D systems are presented first, followed by the 3D variations that evolved from them.

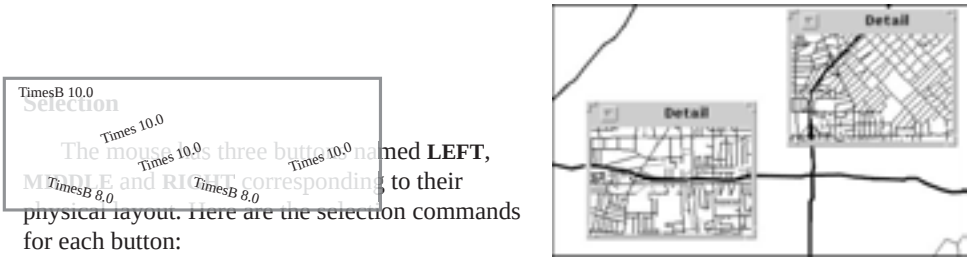
The first Magic Lens systems dealt primarily with manipulating graphics objects (Bier et al. 1993). For example, objects could be rendered differently through a lens by adding a drop shadow, or rendering them in greyscale.



(a) Greyscale and drop shadow lenses.

(b) A local scaling lens.

Figure 2.34: Original Magic Lens demonstrations from the ToolGlass interface of Bier et al. (1993). Images courtesy of Eric Bier.



(a) A Magic Lens showing font information in a text editor. (Image recreated to increase quality).

(b) A Magic Lens showing local detail on a map.

Figure 2.35: General purpose Magic Lens applications of Stone et al. (1994). Images courtesy of Eric Bier.

Complex patterns of shapes could be simplified through local scaling (see Figure 2.34). The notion of using Magic Lenses for enhancing illustrations was developed later by Bier, Stone and Pier (1997).

Stone, Fishkin and Bier (1994) extended the initial work on Magic Lenses by applying them not only to graphics, but to more general purpose applications within the user interface, including text editors (Figure 2.35(a)) and mapping programs (Figure 2.35(b)). Magic Lenses were later applied in purely data-driven applications to specify dynamic queries (Fishkin and Stone 1995), as described in Section 2.2.1.

The Magic Lens concept has also been applied to user interface design

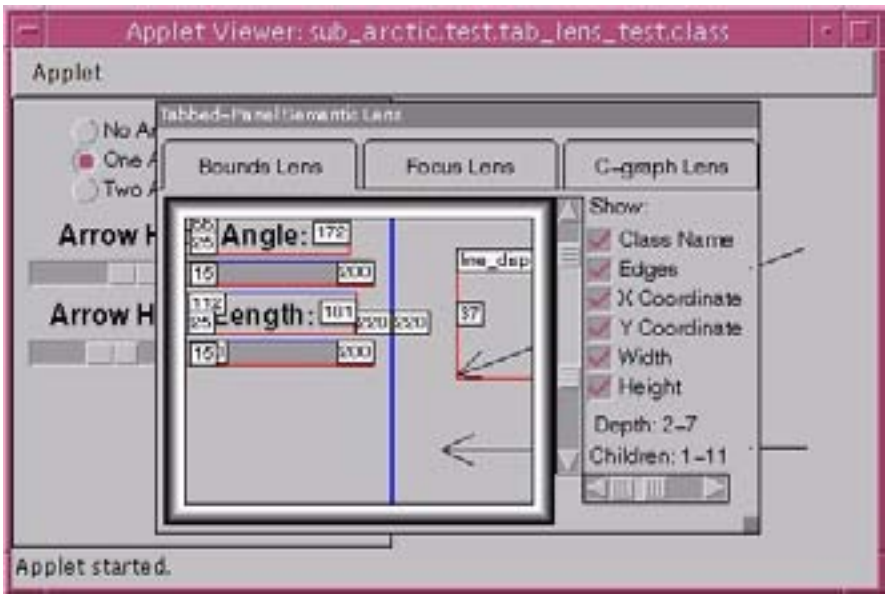
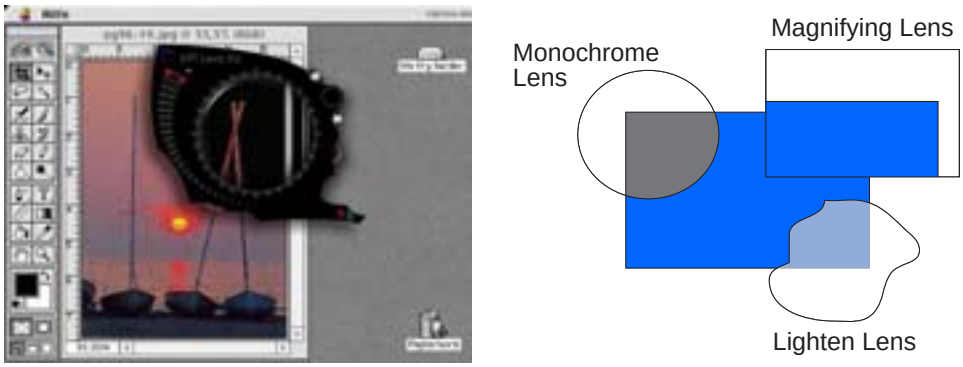


Figure 2.36: Debugging Lens of Hudson et al. (1997). Image courtesy of Scott Hudson.

(Hudson et al. 1997). While a GUI application was running, a debugging lens could be dragged over the various interface components to interactively expose detailed information about their positions, dimensions and overall state (see Figure 2.36). In many instances, such lenses removed the need for traditional debugging printouts and program breakpoints.

Continuing with their original purpose, Magic Lenses have been applied within commercial graphics applications. Kai's Scope is a plugin for Photoshop that lets an image filtering lens be dragged over the workspace in real time (see Figure 2.37(a)). Corel Draw and Macromedia Freehand have lens fill effects which can be applied to shapes so that they change the appearance of shapes beneath them. Three effects available in Freehand are shown in Figure 2.37(b).

In data visualisation, the Sampling Lens of Ellis, Bertini and Dix (2005) intelligently lowers the number of data items in a 2D scatter-plot to reduce clutter introduced by overplotting (see Figure 2.38(a)). The sampling lens was partly motivated by the EdgeLens of Wong, Carpendale and Greenberg (2003). The EdgeLens interactively curves edges of a graph away from the



(a) Kai s Scope in Adobe Photoshop. (b) Lens fill effects in Macromedia Freehand.

Figure 2.37: Magic Lenses in Graphics Applications.

lens centre while leaving nodes in place. This makes it easier to see the nodes while still maintaining edges (see Figure 2.38(b)).

In recent years it has become possible to add more interactivity and dynamic behaviour to websites. Several web technologies now make it possible to implement magnification and Magic Lens effects within hypertext documents on the internet.

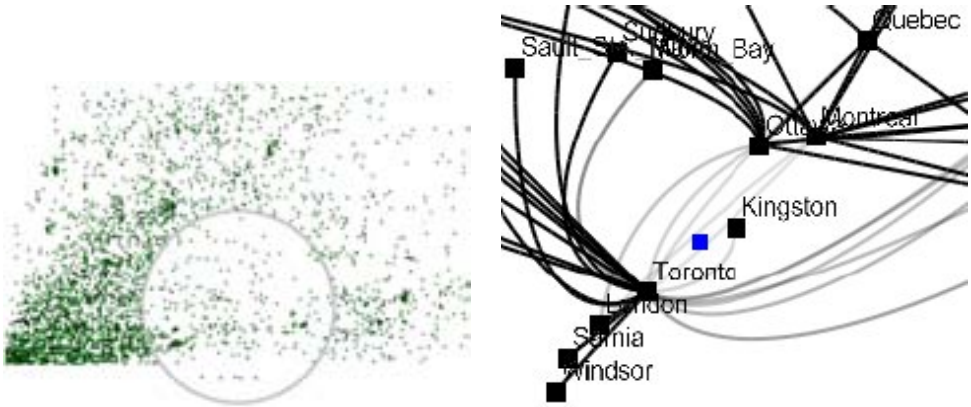
The British Library uses Shockwave to display highly detailed scans of old books. In the case of Leonardo Da Vinci's notebook[3], it is possible to activate a magnifying lens which has the additional feature of flipping the image so that Da Vinci's distinctive mirrored writing is corrected. This is shown in Figure 2.39(a).

The DoHistory website[4] uses a Java applet to present the handwritten diary of Martha Ballard, an American who wrote a diary entry nearly every day from January 1, 1785 to May 12, 1812. The site uses a Magic Lens, shown in Figure 2.39(b), to display the transcribed text of the diary entries while still showing the original handwriting elsewhere.

Industrial Light and Magic produced the special effects for the movie

[3] The British Library, *Turning the Pages*, `http://ttp.bl.uk/collections/treasures/leonardo/leonardo_broadband.htm` (online as of September 2007)

[4] DoHistory, *Martha Ballard's Diary Online*, `http://dohistory.org/diary/exercises/lens/index.html` (online as of September 2007)



(a) The Sampling Lens. Image courtesy of Geoff Ellis.

(b) The EdgeLens. Image courtesy of Nelson Wong.

Figure 2.38: Magic Lenses in Data Visualisation Applications.

Pirates of the Caribbean: Dead Man s Chest. On their website[5] they use Flash to demonstrate some of the post-production techniques used to create the effects. One of their examples uses a Magic Lens to show live video before effects were applied, while the rest of the video shows the view afterward. Figure 2.39(c) shows frames with and without the lens in view. As the video plays, the visitor can move the lens around the frame to see what was originally shot on film, and compare it to what finally ended up in the movie. The lens effect is significantly more compelling than a side-by-side comparison, as it encourages exploration, provides interactivity, and engages the user.

It is also possible to achieve Magic Lens effects without extra plugins. Newer web browser versions support Dynamic HTML and Cascading Style Sheets, technologies that can be used to add interactivity to otherwise static web pages. Examples of magnifying lenses using these techniques have appeared online, and with slight modifications it was possible to extend them to have Magic Lens behaviour (see Figure 2.39(d)).

[5] Industrial Light and Magic, *The Show*, `http://www.ilm.com/theshow/` (online as of September 2007)

(a) The British Library uses a Shockwave presentation to show Leonardo da Vinci s Notebook with a magnifying lens that can flip da Vinci s famous reversed handwriting. Image copyright British Library Board. All Rights Reserved.

(b) The DoHistory website uses a Java applet to show Martha Ballard s diary. The handwritten version is shown as context, and a Magic Lens can be used to investigate words that are difficult to read. Image courtesy of `DoHistory.org`.

(c) Industrial Light and Magic use a Flash movie to show the process of adding computer-generated effects to a film. The Magic Lens reveals that the film looked like before the effects were added in.

(d) This Magic Lens was implemented with only DHTML and CSS and operates in a web-browser without any additional plugins.

Figure 2.39: Web-based Magic Lenses.

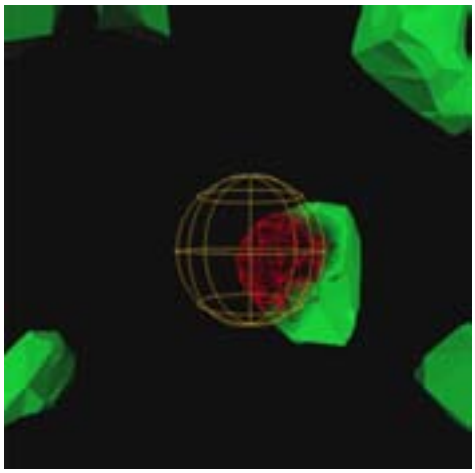
2.3.2 3D Magic Lens Systems

The Magic Lens concept can be extended to three dimensions by considering the input and output regions to be volumes rather than flat surfaces.

The earliest found example of a 3D Magic Lens was the MagicSphere (Cignoni, Montani and Scopigno 1994). This was a volumetric visualisation tool designed to improve graphics performance when rendering large three-dimensional datasets. It consisted of a spherical widget which represented the user's region of interest (see Figure 2.40(a)). Performance was increased by rendering high detail graphics within that region, and low detail elsewhere. Other filter types could be applied, such as a wireframe renderer and a surface interpolator. The MagicSphere operated at the vertex level, resulting in faces being either entirely inside, outside or straddling the border of the widget.

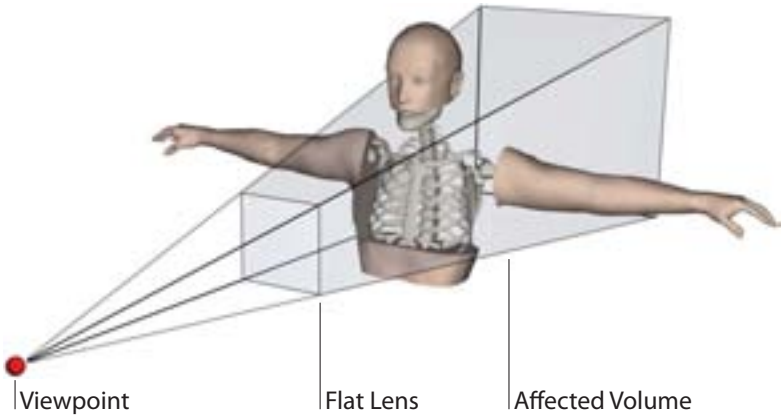
More flexible 3D Magic Lens implementations were provided by Viega, Conway, Williams and Pausch (1996). They introduced two 3D lens variations: flat and volumetric lenses. A flat lens projects a region of interest into the scene by casting rays from the viewpoint through the corners of a lens face (see Figure 2.40(c)). A volumetric lens is a rigid cube that can be positioned within the scene and is viewpoint independent (see Figure 2.40(b)). The operation of the lenses, however, is essentially the same: any 3D geometry falling within a lens region is modified by the lens filter. Both these lens types were implemented using hardware clipping planes, a recently available graphics card feature at the time. By clipping the geometry, these lenses overcame the per-vertex restriction of MagicSphere.

The World in Miniature (WIM) interaction metaphor is a technique in which the user holds a small version of the virtual world in which they are immersed (Stoakley et al. 1995). They can use it from an exocentric viewpoint as a proxy object to interact with the virtual world they are immersed within (see Figure 2.41). The WIM is a dynamic, interactive map of the virtual world. If the virtual environment is considered the input region, and the WIM object is the output region, then the WIM can be thought of as a "reduction" lens that minimises the world to a convenient scale. Furthermore, the authors discuss how different representations of the world could be displayed in the miniature version, which would solidify WIM as a Magic



(a) The MagicSphere of Cignoni et al. (1994).

(b) A volumetric 3D Magic Lens (Viega et al. 1996). Image courtesy of John Viega.



(c) A flat 3D Magic Lens. This image was created based on the description provided by Viega et al. (1996).)

Figure 2.40: Early 3D Magic Lens implementations.

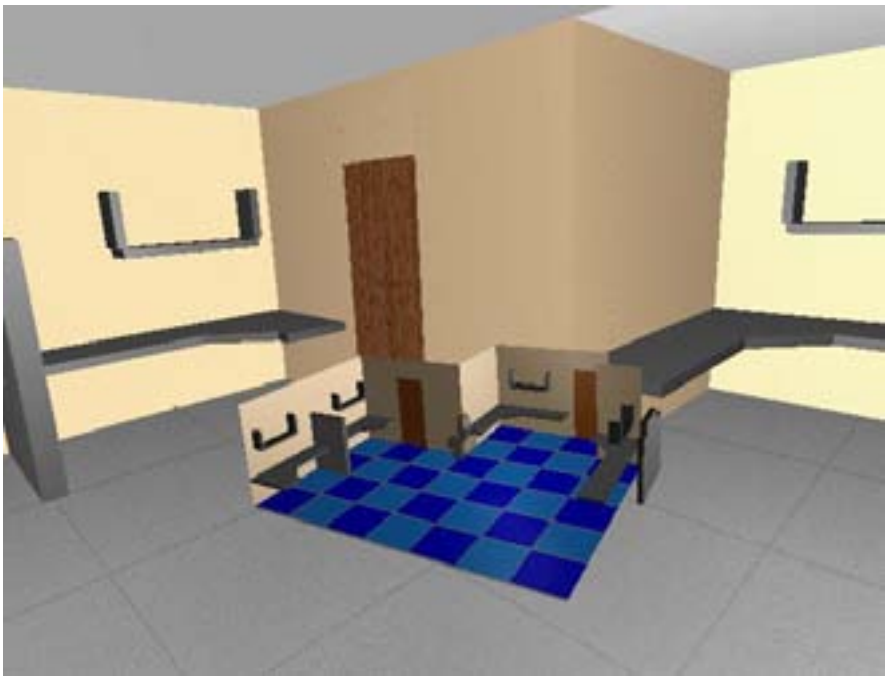
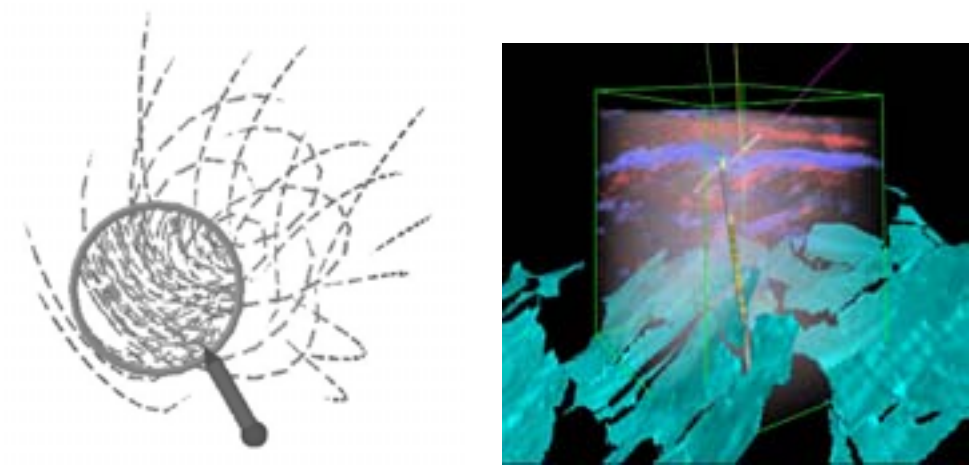


Figure 2.41: World in Miniature (Stoakley, Conway and Pausch 1995).

Lens technique.

The Virtual Tricorder (Wloka and Greenfield 1995) was a general purpose tool for immersive virtual reality. It was based on the multi-purpose handheld "Tricorder" device made famous as an indispensable tool on *Star Trek* films and television shows. The Virtual Tricorder was mapped directly to a six degree-of-freedom (DOF) controller and provided a uniform interface to a number of virtual tools, including a Magic Lens visualisation tool.

Flow visualisation is a type of scientific visualisation in which the patterns produced by flowing fluids (such as air and water) are represented visually for analysis. A common example is the airflow over a vehicle to test aerodynamics. The complex nature of the data and resulting visuals lend themselves to the filtering capabilities of Magic Lenses. In a 3D flow visualisation, Fuhrmann and Gröller (1998) used a volumetric Magic Lens to render dense flow lines within the lens and only sparse lines outside (see Figure 2.42(a)). Fröhlich, Barrass, Zehner, Plate and Göbel (1999) also used a volumetric lens to explore geo-scientific data as shown in Figure 2.42(b).



(a) Magic Lens for flow visualisation (Fuhrmann and Groller 1998). Image courtesy of Anton Fuhrmann.

(b) Volumetric lens for geo-scientific visualisation (Frohlich et al. 1999). Image courtesy of Bernd Frohlich.

Figure 2.42: 3D Magic Lenses for visualisation.

Their lens was controlled by the cubic mouse, a 6 DOF (degree-of-freedom) controller with additional axis manipulators.

A SEAM is a Spatially Extended Anchoring Mechanism, a door that connects two virtual worlds (Schaufler and Schmalstieg 1999). In a virtual world it is represented as a polygon through which the user can both peer and transition. A hyperlink defines the logical location of the remote environment and a transformation matrix defines the relationship between the endpoints. SEAMS are like "magic mirrors" and "wormholes" found in books and movies. They can also be used to implement 3D Magic Lenses by carefully constructing similar local and remote worlds, where the remote world presents the modified view seen through the SEAM/lens.

The SEAMs architecture was used in the "through the lens" model of interaction by Stoev, Schmalstieg and Straßer (2002). The system consisted of a handheld panel and stylus input device used within a semi-immersive virtual environment. The panel presents the user with a Magic Lens type view. The "second world" seen through the lens can be manipulated independently of the primary world. Remote object manipulation is achieved by

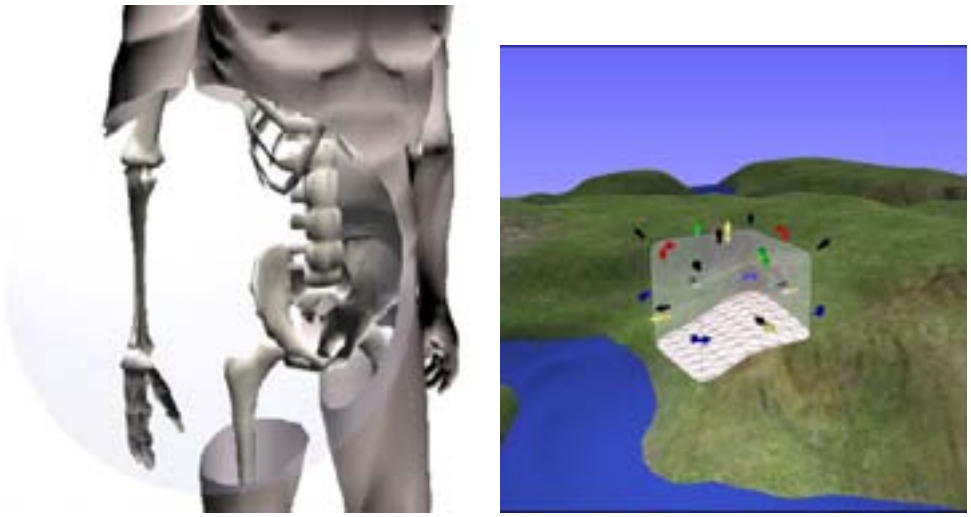
zooming and dragging the world within the lens until the object of interest is conveniently positioned in view. It can then be worked on through the lens using a selection of manipulation tools.

SCAPE (Stereoscopic Collaboration in Augmented and Projective Environments) is a collaborative augmented reality environment based on head-mounted projective display (HMPD) technology (Hua, Gao and Brown 2003). The physical environment is covered in retro-reflective material so that images projected from the user's viewpoint are reflected directly back towards their eyes. A variety of user interface tools were implemented for this system, including a magnifying handheld prop. The authors suggest that this magnifier could be enhanced with some Magic Lens properties, and demonstrated this technique in later work (Brown, Hua and Gao 2003). More recently, they evaluated different lens sizes for search tasks.

A technique related to Magic Lenses was proposed for terrain visualisation by Döllner, Baumman and Hinrichs (2000). Multiple texture layers could be drawn on a terrain model and blended together. In a variation of this blending approach, called Texture Lenses, a special mask texture could be incorporated into the blend to highlight particular geographical locations.

Ropinski and Hinrichs (2004) presented a new rendering algorithm for volumetric Magic Lenses with arbitrary convex shapes. Previous work produced either spherical or box-shaped volumetric lenses. The new algorithm takes advantage of advanced rendering techniques such as shadow mapping, depth peeling and projective texture mapping (see Figure 2.43(a)). Ropinski, Hinrichs and Steinicke (2005) demonstrated Magic Lenses based on these techniques in geographical visualisation scenarios (see Figure 2.43(b)).

There have been several variations on the 3D Magic Lens theme. Virtual mirrors, for example, share many of the characteristics of virtual lenses. The Magic Mirror of Grosjean and Coquillart (1999) mimics the behaviour of a real mirror to aid in the exploration of virtual objects (see Figure 2.44(a)). In the same way that a Magic Lens enhances a magnifier, the mirror metaphor is also enhanced. The image on the mirror can be flipped so that reflected text can be read and the frustum in front of the mirror is clipped so that the mirror is always visible and usable, even when moved within objects. As shown in Figure 2.44(b), virtual mirrors have also been used in medical



(a) New rendering techniques for volumetric lenses. (Ropinski and Hinrichs 2004).

(b) Volumetric GIS Lens (Ropinski et al. 2005).

Figure 2.43: 3D Magic Lenses for visualisation. Images courtesy of Timo Ropinski.

visualisation applications (Bichlmeier, Sielhorst and Navab 2006).

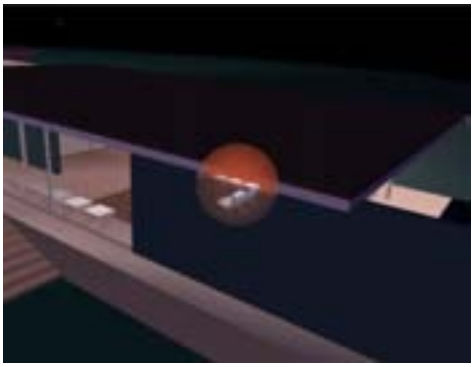
In his Masters thesis, Napari (1999) describes implementation approaches for 3D Magic Lenses and also introduces his own variation called *Magic Lights*. Rather than displaying a modified view of a scene in a lens, a Magic Light projects new information directly into the scene, much like a data projector (see Figure 2.44(c)). The light metaphor is appropriate in that otherwise hidden information is illuminated, and that the effect falls off in the same way as light attenuates over distance.

Spray rendering can be used to produce effects similar to Magic Lenses. Rather than applying a transformation to content seen through a lens, the transformation is applied to content that has been painted by spray particles. This concept was used in the CSpray architecture of Pang, Wittenbrink and Goodman (1995) shown in Figure 2.44(d). CSpray strengthened the metaphor by using a virtual spraycan to represent the tool.

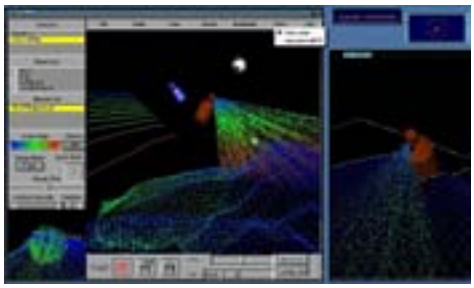
The virtual spray is a particle system, the computer graphics technique used to simulate phenomena such as fire, clouds and water. In a particle

(a) The Magic Mirror of Grosjean and Coquillart (1999).

(b) A Medical Mirror (Bichlmeier et al. 2006). Image courtesy of Christoph Bichlmeier.



(c) A Magic Light showing objects inside a building (Napari 1999).



(d) Spray visualisation with CSpray by Pang et al. (1995). Image courtesy of Alex Pang.

Figure 2.44: 3D Magic Lens variations.

system, particles are ejected from an emitter with a set of parameters that define how they will appear and act, such as their lifetime, initial velocity, mass and colour. Different sets of values are used to produce different effects. As time passes, the system simulates the behaviour of the particles based on environmental parameters, such as gravity and wind, and particle collisions with objects. With spray rendering, when these collisions occur, the particles leave behind a lasting mask surface that defines where the modified visualisation should show through.

2.3.3 Summary of Magic Lens Systems

In this section we have provided a summary of Magic Lens interfaces, which provide a natural way of supporting Focus and Context interfaces. As we have shown, there has been considerable innovation and implementation in lens-based interfaces. However, there have been few user studies conducted to evaluate their effectiveness and no real interface guidelines exist. Most recently, the Magic Lens has been extended into 3D and a variety of interesting variations developed such as the World In Miniature technique. However, once again, there have been few formal user studies conducted or specific interface guidelines developed. This is the research gap that we are seeking to address with this thesis.

2.4 Augmented Reality

Augmented Reality (AR) involves the real time superimposition of computer graphics on the real world (Azuma 1997). An interesting concept in the study of human perception is the Ambient Optical Array (Gibson 1979). If we consider the world around us a complicated system of light reflecting and absorbing surfaces, with a steady stream of light from various sources, then for a particular observation point in space, there is a unique arrangement of visual information completely surrounding the observer. This is the Ambient Optical Array. As the environment changes, and the observer moves about, the array shifts and changes as well. This gives rise to *optical flow*.

Ware (2004) has already related the Ambient Optical Array to computer graphics. He states that "much of the effort of computer graphics can be characterized as an attempt to model the ambient optical array". This means accurately visualising the bundle of light rays that eventually reach the viewer's eye.

Continuing this line of thinking, the goal of computer graphics for augmented reality is to adjust the ambient optical array such that it includes simulated light rays arriving from virtual objects. This is shown in Figure 2.45, where the dotted object is not real, but will appear so to the viewer if it can be accurately represented within the Ambient Optical Array.

Azuma (1997) provides a more concrete definition of AR that is widely

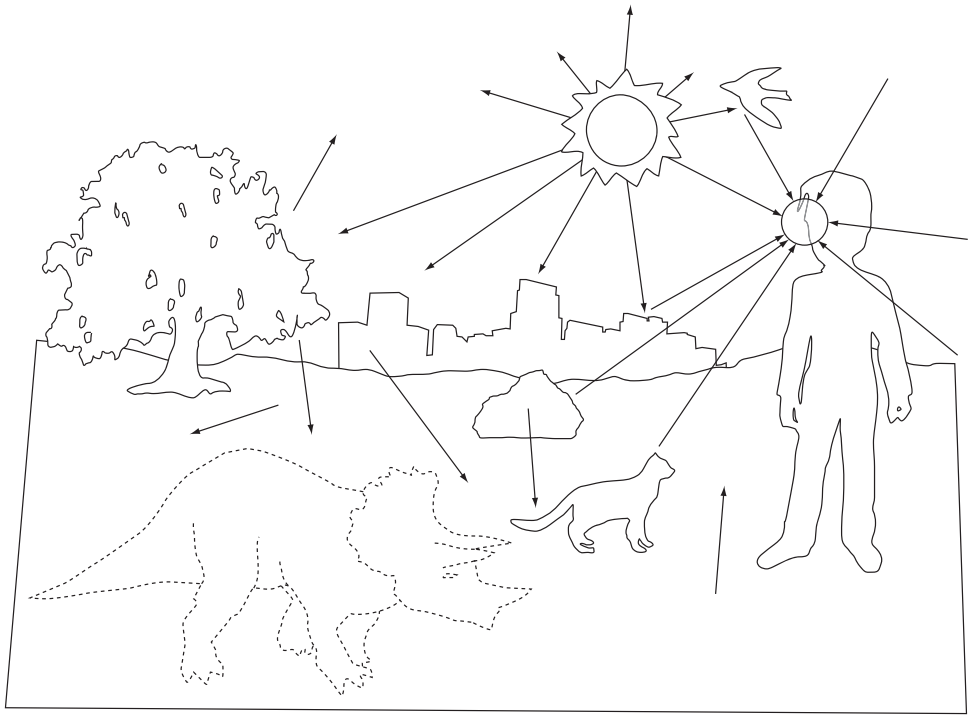


Figure 2.45: Computer graphics can be considered as attempting to accurately visualise the bundle of rays hitting the viewer's eye (the Ambient Optical Array). Augmented Reality can be considered as the addition of virtual imagery, such as the dinosaur in the above image, into the Ambient Optical Array. Diagram adapted from Ware (2004).

accepted in the computer graphics community. This definition states that in an augmented reality system, the following three conditions must be met: the computer graphics must be three dimensional, accurately registered within the real environment and the entire process must occur in real time.

Registration refers to the task of aligning virtual and real objects such that they appear to be connected or collocated. Therefore, the definition precludes interfaces and experiences such as movie special effects, which are generated offline in an time-consuming process, and news broadcast overlays, which simply appear statically in front of the video image, independent of movement in the images themselves.

AR can also be seen in the context of the Reality-Virtuality continuum (Milgram and Kishino 1994). This is a continuum that encompasses all experiences ranging from the completely real to the completely virtual (see Figure 2.46).

Figure 2.46: The Reality-Virtuality Continuum.

A real experience, such as reading a book or going for a walk, requires no mediation by technology. A virtual experience is the inverse; reality is substituted for an immersive computer-generated world. The middle ground between these two extremes is known as Mixed Reality, and includes Augmented Reality and Augmented Virtuality.

Augmented Reality presents predominantly real content, but embeds some computer generated content to aid the user in some way. Augmented Virtuality presents predominantly virtual content, but embeds some real-world elements such as a video feed from an actual location.

2.4.1 The History of Augmented Reality

The first augmented reality system was created by Ivan Sutherland in 1965 (Sutherland 1968). The goal of the research project was to develop the "ultimate display", using a virtual reality headset with an accompanying tracking rig (see Figure 2.47). Sutherland noted that the half-silvered mirrors used in the display allowed the real world to show through, such that the virtual graphics (simple line drawings at the time) could be made to "coincide with maps, desk tops, walls or the keys of a typewriter".

Figure 2.47: Sutherland's "Ultimate Display", 1965.

Through the 1970s and 1980s most AR research was undertaken by the military. For example, the US Air Force Super Cockpit program (Furness 1986) used previous experience with Head-Up Displays (HUDs) in cockpits to project virtual imagery directly on the pilot's helmet and provide an AR experience. Although this research was very successful, it was not generally known outside of the military.

The first use of the term "augmented reality" was by Tom Caudell in an industrial setting at Boeing in the early 1990s. He used the term to describe a head-mounted display system used to train workers to install wire harnesses in aircraft (Caudell and Mizell 1992).

In the late 1990s, AR emerged as a way to enhance live sports broadcasting. An early example is the "FoxTrax" ice hockey puck shown in Figure 2.48(a). The special puck could be tracked within the sports arena and could be highlighted with glow and comet trail effects within the video broadcast (Cavallaro 1997). The "first down" line in American Football broadcasts is

(a) Ice hockey puck with virtual comet trail.

(b) The virtual first down line in American football.

Figure 2.48: Examples of augmented reality in broadcasting.

another example, shown in Figure 2.48(b).

In 1998, the software library ARToolKit was released (Kato and Billinghurst 1999). This computer vision library solved two of the main challenges with AR interfaces, user viewpoint tracking and supporting object-based interaction. It enables the creation of low cost desktop AR systems based on a simple USB camera connected to a commodity desktop PC. ARToolKit was released into the public domain and has since been used by thousands of developers to build a wide range of different AR applications.

Since then the field has grown rapidly with the first dedicated conferences, large-scale, government-funded research projects such as ARVIKA[6], and the first commercial companies such as Total Immersion[7].

2.4.2 Providing an AR Experience

In order to seamlessly add virtual content to the real world, there are a number of different technologies that must be available. First, there needs to be a way to track the user's viewpoint accurately and continuously in real-time so that the virtual graphics can be drawn from that position. Display technology must be used to combine graphics with a view of the real world

[6] ARVIKA, `http://www.arvika.de/www/index.htm`, online as of September 2007.

[7] Total Immersion, `http://www.t-immersion.com/`, online as of September 2007.

and create the AR view. Finally, interaction methods must be used to allow the user to interact with the virtual objects shown in the AR scene. In this section, we describe each of these technology areas in more detail, and then close with a brief description of current AR applications.

Tracking

Tracking is the basic enabling technology for AR, and is required to enable accurate measurement of where the virtual content should be rendered from. Although both AR and immersive Virtual Reality interfaces involve tracking of the user viewpoint, tracking requirements are stricter in AR than VR because of the higher accuracy and precision required to maintain the illusion of a merged real-virtual environment. Lag in a VR tracking system may be frustrating and reduce performance, but lag in an AR tracking system can also cause virtual objects to float behind their correct positions, again destroying the illusion of augmented reality. Therefore, poor or noisy tracking can limit AR's viability in many application areas such as medical, and industrial tasks, where precision and reliability become matters of personal safety.

There are many possible tracking technologies that could be used. Azuma (1997) mentions magnetic, mechanical, inertial and computer vision based systems among others. All have disadvantages and advantages and the ideal system depends on the desired application. For example, for outdoor AR systems a hybrid combination of GPS and inertial/computer vision systems provides a good result (Azuma, Hoff, Neely and Sarfaty 1999) but GPS technology does not work indoors and so is not appropriate for many industrial applications.

In the work described in this thesis, the ARToolKit computer vision library (Kato and Billinghurst 1999) is used extensively. ARToolKit is an open-source tool that uses image processing techniques to calculate camera pose from a single square fiducial tracking marker. ARToolKit calculates camera pose at more than 30 frames per second and to millimetre level accuracy, and so is an appropriate tracking approach for the applications described later in this thesis.

AR Displays

There are a number of different technologies that can be used to present AR and afford different styles of interaction. Traditionally, AR interfaces have used head-worn displays to provide immersive experiences. For example, with an optical see-through AR interface (Azuma 1997) users wear a see-through head mounted display which allows them to view graphics displayed directly on the real world. Other researchers have experimented with handheld AR interfaces based on LCD displays (Rekimoto 1995), PDAs or mobile phones (Henrysson, Ollila and Billinghurst 2005), and even projected AR display where virtual imagery is projected onto real world objects (Bimber, Encarnaçao and Schmalstieg 2003). In our research we use a video see-through AR approach.

Video See-Through Augmented Reality is currently the most common approach to creating an augmented reality interface. This technique will be described in more detail than other techniques, due to its popularity and also because it is the approach used in the experiments and implementations described in the later chapters of this thesis.

A small camera and a head-mounted display mediate the user's view of the world around them. Rather than seeing the world directly with their eyes, the user is actually seeing a video image taken by the camera and processed and manipulated by the computer. The computer is filtering our view of reality frame by frame. The process is displayed in Figure 2.49 and described below.

In order for the computer to embed graphics accurately within each image frame, the software first needs to know precisely where the user is located and how they are oriented within the real environment. It also needs to know the parameters of the camera used to capture the frame, so that aspects like field-of-view and optical distortion of the lens can be accounted for. These parameters are discovered during an initial calibration stage.

Many different tracking technologies can be employed to provide the required position and orientation information. Optical tracking is one approach that has become popular, particularly in research applications, due to its low

cost and the availability of free software libraries such as the ARToolKit (ARToolKit 2001).

Once a frame is captured from the camera and transferred to the computer (which occurs roughly 30 times a second to maintain interactivity) the process of registration can begin. The ARToolKit library accomplishes this task using computer vision techniques to determine the position and orientations of black square markers within the video frame. The ARToolKit computes a 3x4 transformation matrix for each detected marker, which provides six degree-of-freedom tracking.

There are some limitations to the marker-based approach. The entirety of each marker must be visible for the marker to be identified and tracked. Also, lighting conditions can affect the camera's ability to provide a frame suitable for processing.

One approach to solve this problem is to use multiple markers grouped together to provide redundancy. Tracking will be possible as long as one of the markers in the set is visible to the camera. This approach is especially useful with applications that deal with the relationships between markers, which can often occlude each other, because a large grid of markers can often be reliably tracked while another marker is positioned in front.

Although the HMD provides the most immersive augmented reality experience, it is sometimes dispensed with in favour of traditional display devices such as monitors, televisions and projectors. This is particularly the case in settings like museums where a public display requires more people to see an exhibit even if they are not controlling the interactivity, removes health concerns about wearing a headset worn by many others, and removes the downtime introduced by people swapping over and adjusting the display to their comfort. The chance of people suffering motion sickness is also lessened.

Interaction in Augmented Reality

In addition to viewing AR content, users often need to be able to interact with the virtual model as well. To support this there have been attempts to bring familiar 2D user interface components into augmented reality interfaces, such as ARWin of DiVerdi, Nurmi and Höllerer (2003).

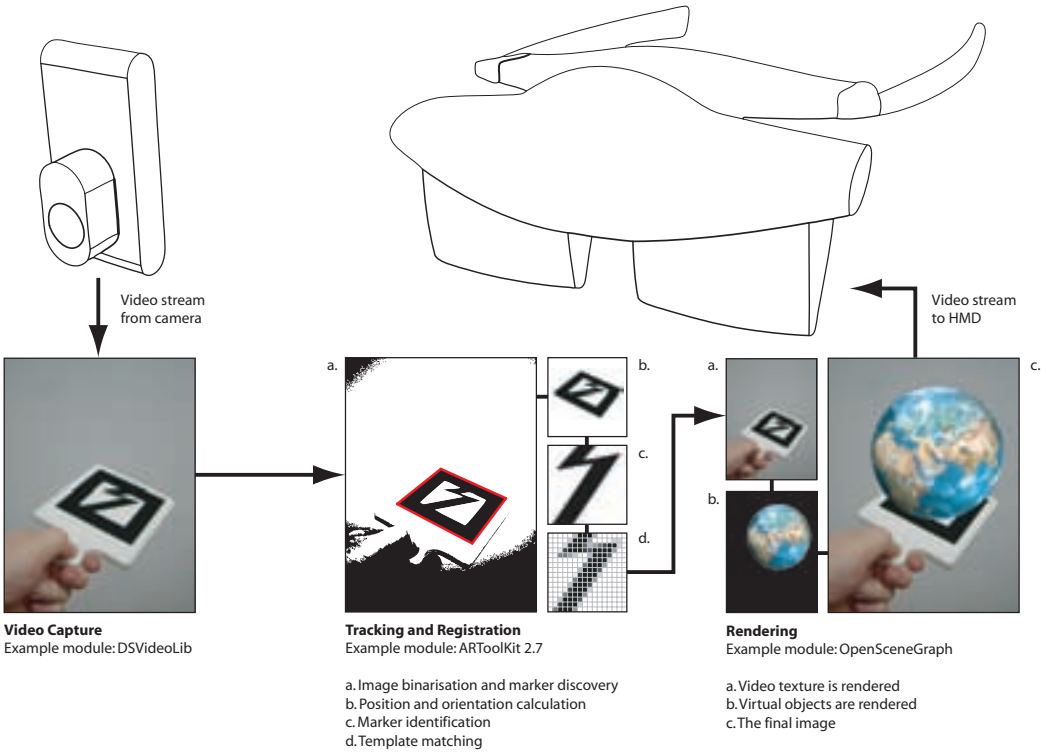


Figure 2.49: The video see-through process for creating an AR interface.

Although familiar 2D interfaces have their place, research into Tangible User Interfaces (TUIs) presents an alternative approach for interaction with AR applications. TUIs use real-world objects as the input and output devices for computers (Ishii and Ullmer 1997). This allows users to interact naturally with the interface components by picking them up, manipulating them with their hands and moving them about. The human hand is capable of fast and precise manipulations so it makes sense to exploit this capability in a user interface.

Examples that demonstrate the benefits of a tangible input approach include the cutting planes of Hinckley, Pausch, Goble and Kassell (1994) and the ActiveCubes project of Kitamura, Itoh and Kishino (2001), both shown in Figure 2.50. Hinckley's work uses a real doll's head to allow the user to specify a cutting plane through CT scan data of a patient's brain. The user does this by holding a tracked piece of perspex against the doll's

(a) Selecting a cutting plane with tangible props. Image courtesy of Ken Hinckley.

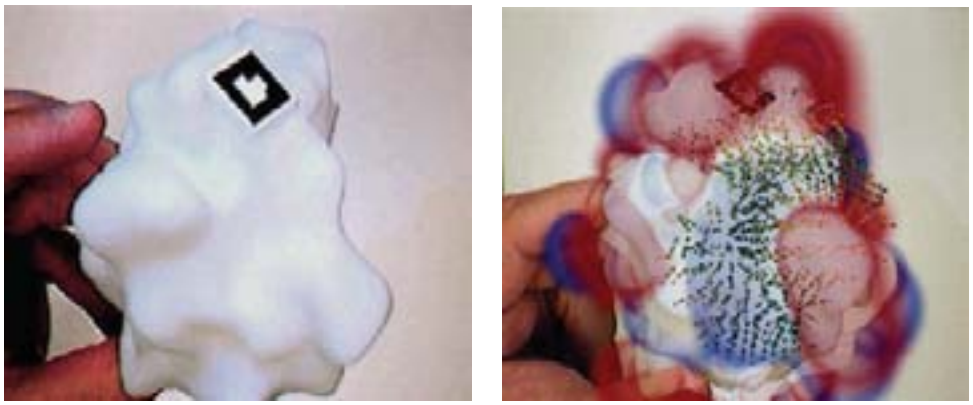
(b) ActiveCubes can be connected together into 3D structures interpreted by the computer.

Figure 2.50: Tangible User Interface examples.

head and viewing the corresponding cutting plane output on the monitor in front of the doll's head. In the ActiveCubes project, cubes with motion sensing electronics in them enable the user to construct simple virtual models and interact with screen-based content. When the user moves the virtual cubes, the corresponding virtual model on the screen moves in the same way, facilitating very intuitive manipulation of the virtual content.

Tangible interfaces are extremely intuitive to use because physical object manipulations are mapped one-to-one to virtual object operations, and they follow a space-multiplexed input design (Fitzmaurice and Buxton 1997). Another benefit of a TUI is that it naturally supports sharing and collaboration. The principles of TUIs can be coupled with AR's display capabilities in an approach referred to as Tangible Augmented Reality (TAR) (Kato, Billinghurst, Poupyrev and Tetsutani 2001). In a TAR interface, virtual objects are registered with physical objects and the virtual "shadow" object is manipulated by interacting with the real physical object.

A good example of TAR is the overlay of virtual information on physical models of molecules, such as a representation of the electrostatic field (Gillet, Sanner, Stoffler and Olson 2005) (see Figure 2.51). In this case the user can hold a model of a complex molecule in their hand and see a virtual overlay



(a) Without augmentation. (b) With augmentation.

Figure 2.51: Tangible augmented reality molecule visualisation. This illustration depicts the superoxide dismutase enzyme with and without AR electrostatic field augmentation. Image courtesy of Arthur J. Olson, The Scripps Research Institute, copyright 2005 TSRI.

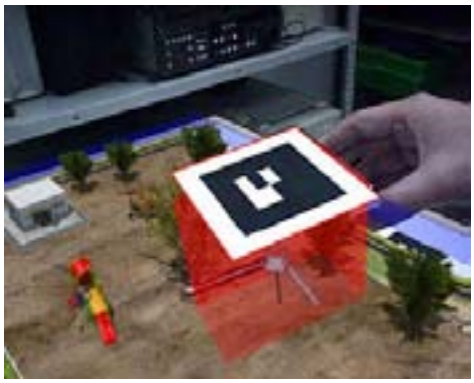
of electrostatic field on the model. This enables the user to easily view any part of the field simply by rotating the real model around.

Other examples of application of the TAR metaphor include the MagicBook transitional interface (Billinghurst, Kato and Poupyrev 2001) (Figure 2.52(a)), the Magic Cup urban design tool (Kato, Tachibana, Tanabe, Nakajima and Fukuda 2003) (Figure 2.52(b)), the Tiles application for virtual cockpit layout (Poupyrev, Tan, Billinghurst, Kato, Regenbrecht and Tetsutani 2001) (Figure 2.52(c)) and the MagicCube edutainment application of Zhou, Cheok, Pan and Li (2004) (Figure 2.52(d)).

2.4.3 Usability Testing in Augmented Reality

Once AR applications have been designed it is important to evaluate interaction techniques using formal usability testing. User studies with AR interfaces generally deal with perception, interaction and collaboration issues (Billinghurst 2001). Drascic and Milgram (1996) compiled a list of 18 perceptual issues relating to AR that need to be considered when attempting to create immersive virtual environments that evoke presence. These issues include providing the user with a sufficient field-of-view and appropriate depth

(a) MagicBook. Image courtesy of Mark Billinghurst.



(b) The Magic Cup. Image courtesy of Hirokazo Kato.

(c) Tiles. Image courtesy of Mark Billinghurst.

(d) Magic Cubes. Image courtesy of Adrian David Cheok.

Figure 2.52: Example tangible augmented reality interfaces.

cues.

One important perceptual cue is Presence, which is defined as the subjective perception that an experience is not artificially created. Often the goal in AR systems is to maximise the sensation of presence through accurately placing virtual objects in the real world and rendering them convincingly.

Bowman and Hodges (1999) state that many interaction techniques get little further than an initial prototype for demonstration purposes and receive no formal evaluation. Since then, however, there have been several examples of AR interaction experiments. For example Mason, Walji, Lee and MacKen-

zie (2001) performed an experiment in which users had to reach and grasp objects both with and without visual and haptic feedback. They discovered that movement time was only correctly modelled by Fitts' law when haptic feedback was available. This indicates that such feedback is an important factor in increasing performance in virtual environments (Billinghurst 2001). At a higher level, Haniff and Baber (2003) evaluated the usefulness of augmented reality in an assembly task. AR is appropriate for such tasks because it can link physical components to virtual guides and instructions. The study suggested that using AR reduced cognitive load compared to paper-based instructions, but it was found that task completion time was slower with AR. This, however, was attributed to cumbersome equipment, rather than AR itself.

There have also been several usability studies conducted with collaborative AR interfaces. One interesting example is the work of Kiyokawa, Takemura and Yokoya (1999) on shared virtual environments (SVEs) and shared augmented environments (SAEs). Kiyokawa's interface supports seamless switching so that each user can smoothly transition to their partner's current scale. This allows each user to explore the environment as they choose, but also to quickly and easily return to a common scale at which video see-through AR is restored. Collaboration is enhanced when users can really see each other rather than avatars because they can pick up on the physical cues and body language people use. The study presented users with a collaborative design task and found that users preferred AR and completed the task faster than in the VR setting.

2.4.4 Applications for AR

Augmented Reality interfaces have been applied in a broad spectrum of domains such as education, engineering and entertainment. The following list provides some representative examples from some key areas.

- Medical

  Using augmented reality for real-time visualization of tactile health examination (Nikishkov and Tsuchimoto 2007)

Virtual Reality and Augmented Reality in Digestive Surgery (Soler, Nicolau, Schmid, Koehl, Marescaux, Pennec and Ayache 2004)

- Industrial and Architectural

  ARVIKA - Augmented Reality for Development, Production and Service (Friedrich 2002)

  CyliCon: a software platform for the creation and update of virtual factories (Navab, Cubillo, Bascle, Lockau, Kamsties and Neuberger 1999)

  Visualization of construction graphics in outdoor augmented reality (Behzadan and Kamat 2005)

  Augmented Reality: An Application for Architecture (Tripathi 2000)

  ARVino - Outdoor Augmented Reality Visualisation of Viticulture GIS Data (King, Piekarski and Thomas 2005)

- Entertainment and Education

  Motivations for AR Gaming (Nilsen et al. 2004)

  Augmented Chemistry (Fjeld, Juchli and Voegtli 2003)

  Using Augmented Reality for Teaching Earth-Sun Relationships to Undergraduate Geography Students (Shelton and Hedley 2002)

  Archeoguide (Vlahakis, Karigiannis, Tsotros, Gounaris, Almeida, Stricker, Gleue, Christou, Carlucci and Ioannidis 2001)

As this list shows, AR applications are beginning to move from the research environment into real commercialisable applications. However, before AR applications are widely used there is a need for the development of intuitive methods for interacting with the AR content. Research such as that contained in this thesis helps fulfil that need.

2.4.5 Summary of Augmented Reality

In this section we have provided a brief overview of AR technology, interaction techniques and current applications. As can be seen, AR is still in its infancy as a research field, but a lot of important work is being undertaken on enabling technologies such as tracking (particularly 3D optical tracking) and displays. In our research we use a video see-through head mounted display with a tangible AR interaction metaphor in an AR lens application. We also plan to evaluate our work using lessons learned from earlier AR usability studies. The wide range of current AR applications show that AR is slowly pervading our lives through sports broadcasting, handheld devices, and AR-capable digital cameras. Research such as that undertaken with this thesis will enable AR techniques to become more common.

2.5 Summary

In this chapter we have reviewed related research that forms that basis of our research. The main focus of this thesis is to investigate the use of Magic Lenses in Augmented Reality. We initially reviewed the use of Focus and Context in desktop interfaces. These Focus and Context interfaces demonstrated how embedding a region of focus in the context of a broader information display enabled users to have a greater understanding of the dataset. Magic Lens systems are a particularly useful way of providing Focus and Context and we reviewed both 2D and 3D Magic Lens systems.

After detailing this screen-based research, we also provided an overview of Augmented Reality technology, particularly AR interfaces based on head-mounted displays. One of the most useful AR interaction metaphors is that of Tangible AR which combines TUI input techniques with AR graphics display.

Chapter 3

Fundamentals of Magic Lenses

This chapter provides a deeper analysis of the Magic Lens concept. Physical lenses provide the metaphor on which virtual lenses are based, which in turn are the motivation for Magic Lenses. The terminology used to describe physical lenses intersects with the terminology used to describe computer graphics. In optics, the terms *virtual image* and *real image* have specific definitions based on the convergence or divergence of optical rays. In computer graphics, *virtual* has become synonymous with *computer-generated*. In addition, a physical converging lens produces a *real image*, although the term *real* is also overloaded when discussing AR. The definitions employed in this thesis are drawn from a computer graphics perspective.

The components of a Magic Lens and the properties that make it a valuable tool are examined in Section 3.1. One property, the ability to combine multiple Magic Lenses to produce composite effects, is explained in detail. The ways in which the properties of Magic Lenses can be exploited to add capabilities to the user interface are explored in Section 3.2.

3.1 Properties of Magic Lenses

As described in Section 2.2.2, lenses in a user interface are a form of multiple view system that employs a spatial separation between views. The unique feature of a lens is its embedded relationship with its parent view. The key components of a Magic Lens, as defined by Bier et al. (1993) are the Output Region, the Lens Border, the Filtering Operation and the Input Region.

- The Output Region defines the area of the workspace that the Magic Lens will occupy.

- The Lens Border is the outline of the Output Region, and is the boundary between the Focus view and the rest of the workspace (Context).

- The Filtering Operation is the transformation that generates the visualisation in the Output Region.

- The Input Region is the area of the underlying workspace that acts as the source for the Filtering Operation.

The Input Region is therefore defined by the Output Region, which is typically controlled by the user through the resizing and repositioning of the Lens Border. With respect to these components, Figure 3.1 shows three types of lens structure: a magnification lens, an offset lens and a Magic Lens. Logically, a Magic Lens is a generalisation of all other lens types, as the Filtering Operation can be designed to simulate any other lens type. Therefore, all other lenses are simply specific instances of Magic Lenses. The remainder of this section describes five significant characteristics of Magic Lenses: explicit area of focus; maintaining context; safe exploration; cross application filters; and multiple simultaneous filters. These characteristics imply ways in which a user interface can make use of Magic Lenses.

Explicit Area of Focus

The lens is separated from the rest of the workspace by its border, and can be thought of as a sub-workspace in which a different state and set of constraints apply. The lens is an explicit area of focus where the user can configure their view for a particular purpose, and the interface can tailor itself accordingly. For example, placing the lens over a particular building on a map indicates interest in that building, so the interface can dedicate more screen space to presenting information relating to that building.

Some interfaces use the mouse cursor for this purpose. Items near or under the cursor are treated as more important. Examples of this type of behaviour include tooltips and mouse over effects. Although useful, these techniques require the mouse cursor to dwell over items of interest. The cursor is a highly dynamic interface object. It flits between tasks whereas

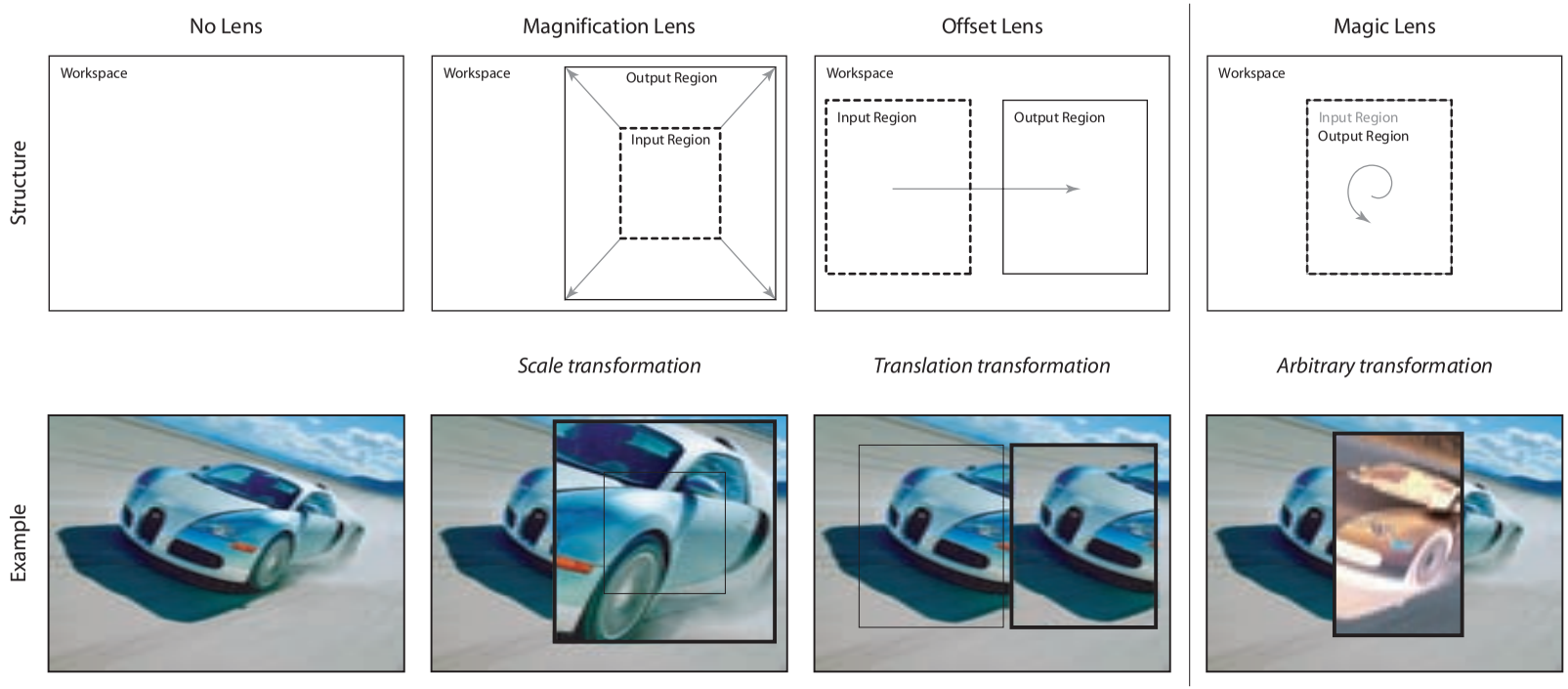


Figure 3.1: The structure of different lens types.

a lens remains where it is put until moved, and continues to define an area of interest for the user, regardless of where the user has subsequently moved the cursor.

Maintain Context

The workspace outside the lens is typically (although not necessarily) unaffected by changes within the lens, and can therefore provide the user with context as they manipulate their area of focus. Section 2.1 described a wide range of techniques that aim to provide the user with a measure of both detailed information and contextual awareness.

Kosara, Miksch and Hauser (2002) identify a class of focus and context technique they call "dimensional" that addresses access to different layers of information. Here the problem is no longer a lack of display space, but rather the best way to visualise collocated information. A Magic Lens can be used in such a situation to view additional layers of information while maintaining a default view as context.

Safe Exploration

Stone et al. (1994) describe the concept of *safe exploration* as a way to protect users from risk when experimenting within a user interface. This safety is often provided by an "undo" command, but can also be provided with a Magic Lens. A Magic Lens can maintain its own state information and operate as an independent and alternative view into the information space. Operations applied within the lens can be limited to altering only the lens's private state. In this way, the user can experiment with different operations and immediately see the effects without worrying about corrupting their previous work.

Even in the presence of a robust undo system, the idea of safe exploration through lenses remains attractive due to the other benefits of lenses, such as being able to compare changes within the lens with the rest of the workspace. Of course, a mechanism for fusing the state of a lens with the rest of the workspace is required so that once a promising result is discovered the operation can be applied globally.

Cross-Application Filters

There is the possibility that the same lens could be used between quite different applications (Stone et al. 1994). If the underlying semantic intention of the lens can be defined, then different applications can interpret this and present their information to the lens in a suitable way. For example, if the point of the lens was to highlight schools, then this could be done when the lens was passed over a city in a mapping application, as well as when the same lens was passed over an accompanying text document.

Multiple Simultaneous Filters

As described in the taxonomy of Focus and Context techniques in Section 2.2, many different views of the same underlying model may be presented in the user interface. Multiple independent Magic Lenses can be active at once, each allowing the user to investigate a different aspect of the model or to pursue multiple avenues of investigation. When multiple filters are used in this way, the possibility of overlap arises. Intuitively, when two lenses overlap, their filtering operations should combine within the intersection area. In their original Magic Lens research, Bier et al. (1993) proposed three techniques to achieve Magic Lens composition. Each technique has associated advantages and disadvantages making it more or less appropriate for certain applications. These techniques are described below, along with a fourth technique, Delegation, proposed later by Fox (1998).

Recursive Ambush is a Magic Lens composition technique that modifies the behaviour of the graphics renderer at runtime. Recursive Ambush can be used when the Model is being visualised through a set of graphics language procedure calls. Each Magic Lens acts as a specialised interpreter for the graphics language, allowing the lens to override the calls to produce custom effects (see Figure 3.2).

Model-In, Model-Out (MIMO) creates a copy of the incoming data Model, modifies it in some way, and makes it available to lenses above (see Figure 3.3). For example, the basic Model might be a collection of 2D points

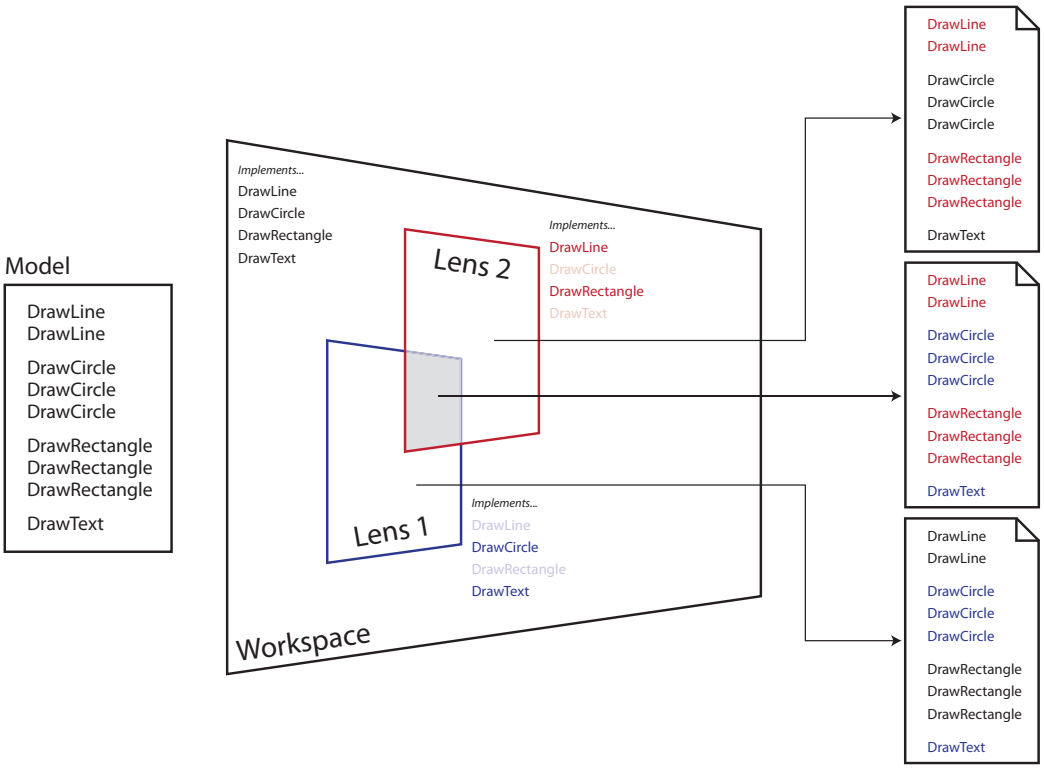


Figure 3.2: The Recursive Ambush technique. Each lens can modify the actions of particular graphics rendering calls. On the left of the above diagram is the list of calls used to generate the visualisation. In the middle, each lens overrides particular calls (shown in bold). On the right, the final code listings are shown for each region of the workspace, colour-coded to show which calls are implemented by which lenses (red and blue), or if they are unmodified (black).

specified as x, y coordinates. An overlapping lens might augment that model with edges connecting certain points, and pass it to another lens which adds labels to the edges in the model. The resulting Model is significantly more descriptive than the original one.

Reparameterise and Clip changes rendering parameters when drawing a Magic Lens region. When rendering primitive objects there are typically a collection of parameters that determine the appearance of the resulting image. For example, lines can have various thicknesses, colours, end-styles,

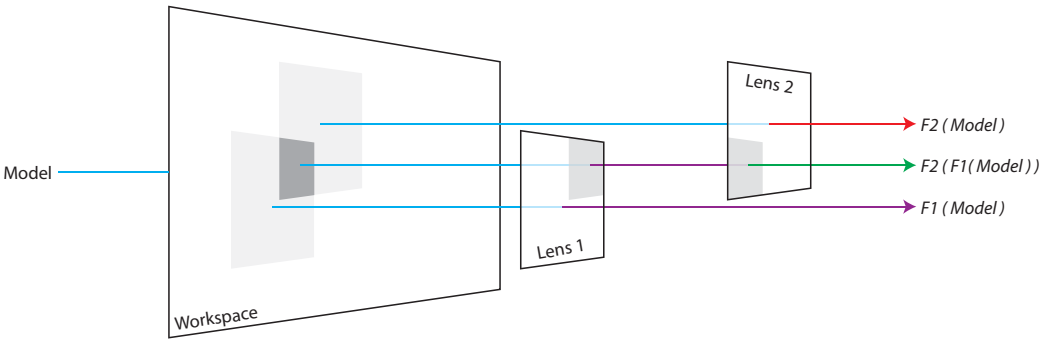


Figure 3.3: The Model-In, Model-Out (MIMO) technique begins with an initial data model to be visualised. The model is passed to lenses within the workspace, which can apply a function (F) to the model to produce a new visualisation. If a lens overlaps another lens, then the model from the underlying lens is passed upwards to the overlapping lens. In this way, composition occurs where lenses overlap, leading to a series of transformations of the original model. (e.g. *F2(F1(Model))*)

dashes, and so on. The values for these parameters can be manipulated to produce new variations. In the Reparameterise and Clip method, each lens can override the current rendering parameters, and redraw the visualisation, clipped to the lens region. This approach is shown in Figure 3.4, where the workspace has one set of rendering values, the lenses have different values, and the intersection of the lenses is a mixture of values, with values from the highest lens in the stack taking precedence.

Delegation is a Magic Lens composition technique proposed by Fox (1998). Software engineers may be more familiar with the terms *Wrapper* or *Decorator* to describe this approach. When a Magic Lens is passed over an object, a new instance of a wrapped object is instantiated. The type of wrapped object created depends on the purpose of the lens. When the view through the lens is generated, it is the new wrapper object that drives the visualisation and handles input from the user. Both these processes, visualisation and interaction, can be handled in a modified way by the wrapper compared to the original object.

In the case of multiple simultaneous lenses overlapping, an input event

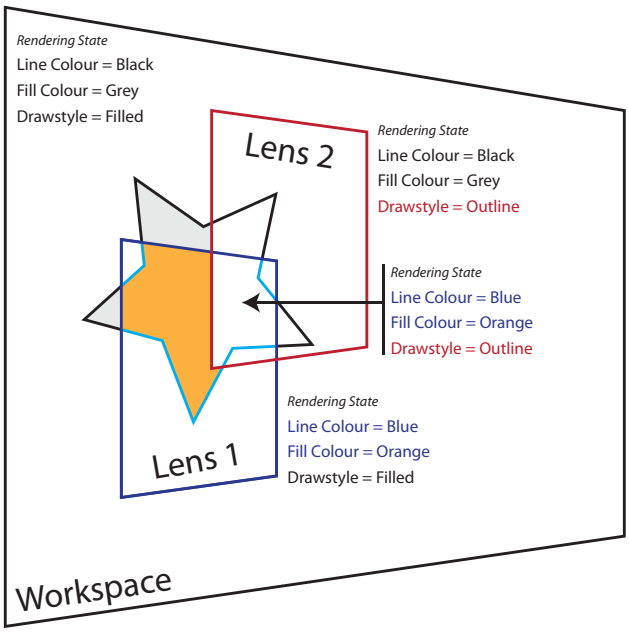


Figure 3.4: In the Reparameterise and Clip technique, each lens can manipulate the values of the renderer's parameters. This allows a lens to modify the appearance of objects. Where two lenses overlap, the two sets of rendering parameters are combined, with the values from the highest lens in the stack taking precedence.

passes down through the stack of lenses, potentially being modified on the way, until it reaches the workspace layer where objects exists. If the event "hits" one of these objects, the lens stack is searched for the first wrapped version of the hit object, starting with the first lens the event encountered. The wrapped object then handles the event. This process is illustrated in Figure 3.5.

3.2 Analysis of Magic Lens Applications

Here we present some practical applications of Magic Lenses. From a thorough survey of Magic Lens research, a classification of application types has been derived. We distinguish between four general types of Magic Lens filtering operation, described in the following sections. A Magic Lens may be used to present:

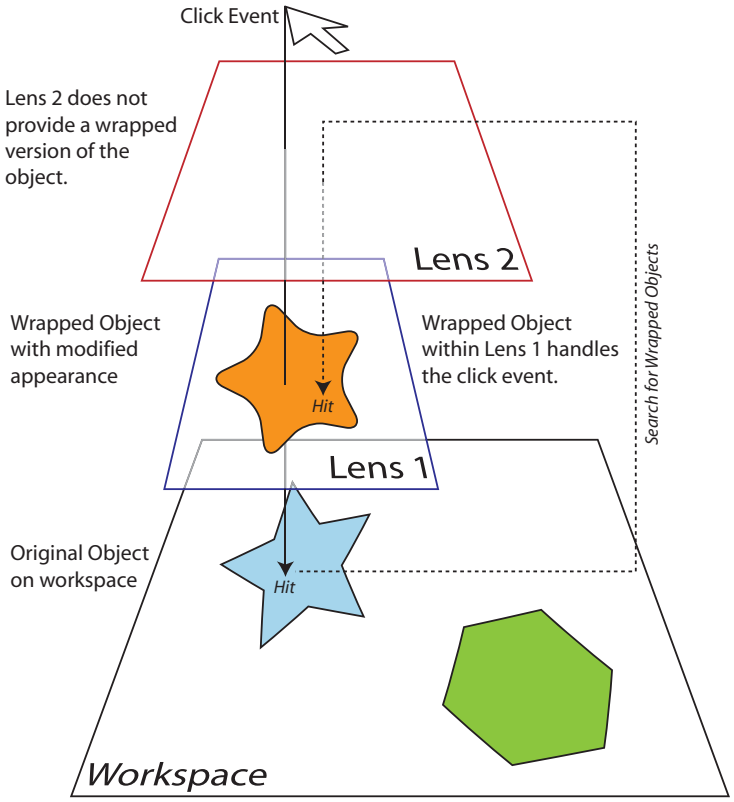


Figure 3.5: Delegation is a composition technique based on the *Wrapper* approach from object-oriented programming. Diagram adapted from Fox (1998).

- Past, Present and Future States
- The Results of User Defined Queries
- Information at Different Levels of Detail
- The Results of an Algorithm

Past, Present and Future States

In some visualisation tasks, users are interested in the way data changes over time. A common way to visualise such data is to present an animation where each frame is a representation of the data for a specific time.

A Magic Lens can be used to provide an alternate visualisation, in which a sub-region of the view shows data from another time offset. This type of visualisation has been investigated in the "Temporal Magic Lens" research conducted by Ryall, Li and Esenther (2005). Here, Magic Lenses were used to simultaneously visualise video footage from different times (see Figure 3.6). Using lenses allowed the user to define both spatial and temporal queries with the same tool.

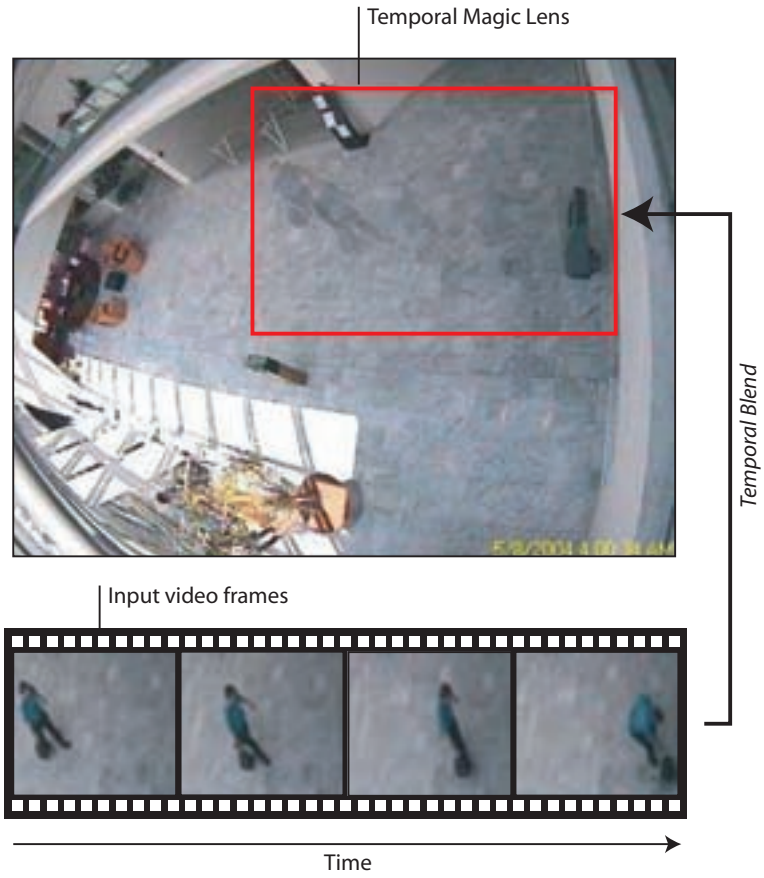


Figure 3.6: This diagram illustrates the Temporal Magic Lens system of Ryall et al. (2005). A set of video frames are composited to create a single view within the lens.

The Results of User Defined Queries

This capability allows the user or system to select, based on any criteria, what information is shown within the lens area. This capability encompasses dimensions other than time which is covered by the first capability.

Consider a real estate scenario, such as the one described by Williamson and Shneiderman (1992), in which the user wishes to suppress all candidate properties that fall outside their price range and do not meet their requirements of three bedrooms and a garage. These requirements form a query where the attributes of price, number of bedrooms and "has garage" are tested for each candidate property within the lens region. Those properties that do not meet the criteria are hidden.

Information at Different Levels of Detail

This capability allows the quantity of information presented to the user to be tailored by the application. Sometimes less information will provide the user with a clutter-free overview, and sometimes targeted information will be required in order to achieve a task.

As explained in Section 2.2.1, semantic zooming is a common way to provide different levels of detail. Semantic zooming couples magnification with information density. As the user zooms in, their view covers a diminishing amount of the information space within the same display space. The resulting extra room can be utilised for more detailed information. This technique can easily be applied within a magnifying Magic Lens.

Sometimes lower detail is required for more practical reasons. Perhaps rendering a 3D scene at maximum resolution is not possible at interactive frame rates unless the view is restricted to a small area. In such a case, low detail can be used over the majority of the workspace, and high-detail, interactive viewing can be maintained within a Magic Lens.

The Results of an Algorithm

Often the user wants to generate new information by applying a transformation to the underlying data. This differs from the capabilities described

thus far in that new data is created rather than existing data being carefully selected.

This capability allows the user to explore various "what if?" scenarios. For example, what effects on the ecosystem would there be if the average temperature in this region was one degree higher?

The user may be interested in information that needs to be calculated in real time. For example, distances from individual houses to the selected school. This information would need to be recalculated whenever the selection changed, and may be a simple Euclidian measurement, or a more complicated network algorithm that takes roads into account.

In graphics applications, this capability could be used extensively in previewing effects such as blurring and edge-detection. These effects apply mathematical transformations to the colour values in an image to generate new colour values. Many such operations take parameters that the user can change at will. Each time a parameter is changed, the algorithm must run again.

3.3 Summary

In this chapter we have examined the fundamental concepts of Magic Lenses. It is important to be aware that although these tools are described as lenses, there are differences in the precise definitions used in physical optics and computer graphics.

The five key characteristics of Magic Lenses emphasise their ability to provide Focus and Context.

- Explicit Area of Focus: A Magic Lens can be used to indicate to the interface the region that is of most interest to the user, relieving the busy mouse cursor of this responsibility and, in contrast, providing stability.

- Maintain Context: A Magic Lens is an ideal tool for interactively viewing layered information. As new layers are enabled within the lens area, the rest of the workspace remains constant, providing context.

- Safe Exploration: A Magic Lens can provide safe exploration by allowing the user to experiment with operations without risk.

- Cross Application Filters: The same Magic Lens can visualise information from different applications, which allows a single lens to perform many roles, enabling the viewer to pursue a single coherent path of investigation rather than multiple related paths in parallel.

- Multiple Simultaneous Filters: Many Magic Lenses can be involved in a coordinated analysis. This introduces the possibility of overlap between filters. Four composition techniques were presented.

Four application areas were identified as representing the main types of filtering used in prior Magic Lens research. Magic Lenses are often used to display information for a particular region at a different time offset from the rest of the workspace. More general queries that operate on dimensions other than time are also common. A particular case is a Level of Detail Magic Lens that increases or decreases the amount of information in the lens view. Finally, the filtering that occurs in the lens may be controlled automatically by an algorithm.

Chapter 4

2D Magic Lens Evaluation

4.1 Introduction

Dense information displays can benefit from filtering that reduces the complexity of the view. Usually filters operate over the entire view at once, which will be referred to as *global filtering*. Magic Lens filters can operate in a secondary view embedded in the main workspace. This approach allows parts of the workspace to be visualised in different ways. When used in this manner, a Magic Lens can be thought of as a *local filter*.

This chapter describes an experiment in which local filtering is evaluated against global filtering and a control case that provides no filtering. The evaluation involves a map analysis scenario, in which participants are required to investigate locations within real-world GIS data.

Many systems containing 2D Magic Lenses have been implemented and demonstrated in the past, a complete survey of which was given in Section 2.3.1. However, evaluation of these interfaces has been minimal. This experiment is a first step in exploring the cases in which Magic Lenses can increase user performance and also aims to elicit subjective feedback from users.

4.2 Scenario

Magic Lenses are a Focus and Context technique as explained in Chapters 2 and 3. This experiment evaluates the concept of Magic Lenses by applying it within a specific scenario: to aid users in an analysis task within a Geographic Information System (GIS).

A GIS is a computer system that integrates all the operations required to analyse, store, edit and display information relating to geographic locations.

Examples of such information include satellite photography, political boundaries, natural features, agricultural and economic data, transportation and weather. These sources of information are managed within the GIS as layers, as shown in Figure 4.1. Layers can be manipulated, overlaid and combined within the GIS.

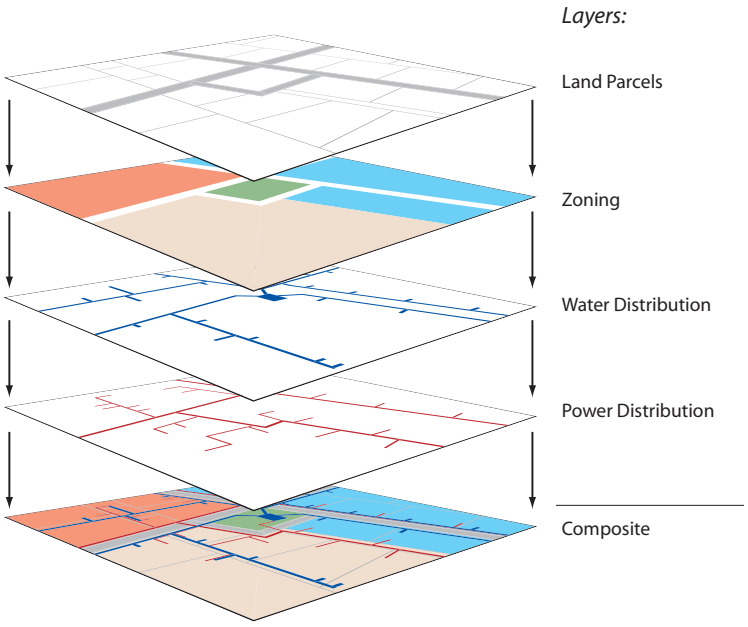
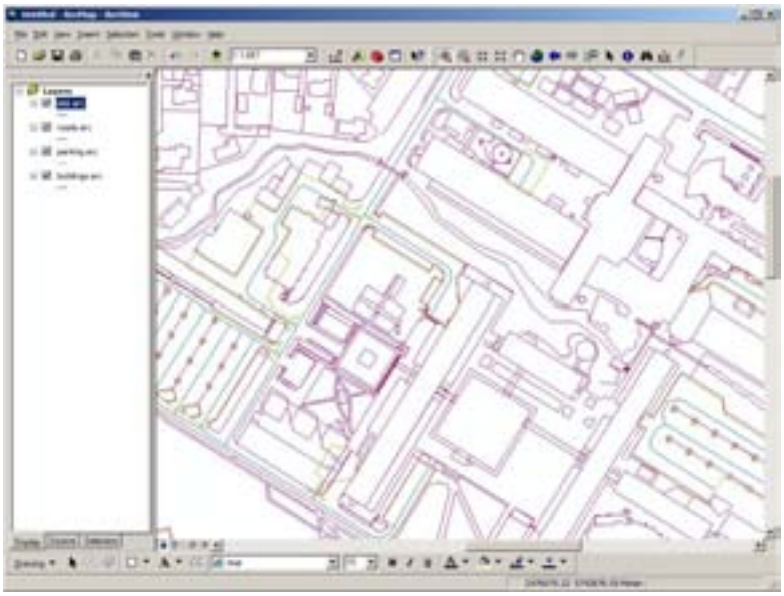


Figure 4.1: Layers within a GIS can be overlaid to build up complex maps.

The underlying motivation for GIS is to uncover useful information within vast geospatial datasets. This is an application area where Magic Lenses may provide significant improvements over existing techniques. Magic Lenses are suited to GIS because of the potentially massive amounts of data that must be managed in such systems and the inherently spatial and visual nature of the data.

Figure 4.2(a) illustrates the layout of ArcView, the most popular professional GIS software package available. Like most GIS packages, the interface is designed to deal with layers of geographical information. Interfaces like this provide tools to manage these layers and perform operations on them. Google Earth^{TM}, shown in Figure 4.2(b), is a consumer-level tool for brows-



(a) ArcView is the industry standard GIS.

(b) Google Earth™ is popular among enthusiasts.

Figure 4.2: Professional and consumer GIS packages. Both employ layers to manage the massive amounts of information that is available.

ing geospatial information. Although these two products represent different ends of the GIS spectrum, their interfaces are remarkably similar. Both present a list of layers, where each layer can be hidden from the main view using a checkbox.

Magic Lenses have previously been proposed as useful tools in the GIS domain. Stone et al. (1994) demonstrated a number of compelling examples of using Magic Lenses to highlight particular geographic features (see Section 2.3.1), and more recently, Kalghatgi, Burgman, Darling, Newbern, Recktenwald, Chin and Kong (2006) have carried out a pilot study indicating that Magic Lenses have advantages over global filtering. There has also been interest in using 3D variations of Magic Lenses in geo-visualisation tasks as described in Section 2.3.2 (Ropinski et al. 2005).

4.3 Experimental Design

A group of users were each asked to carry out map analysis tasks using three different GIS interface designs. Each presented layers of data points overlaid on a map. The first interface showed all layers at once. The second interface allowed each layer to be independently toggled on or off. The third interface also permitted layer toggling, but only with a Magic Lens which could be moved around the map. The tasks required the user to look at a map and count up the total number of points that matched the given search criteria.

The user's speed and accuracy with each interface were recorded, providing a way to quantitatively evaluate which interface performed best. In this section the design of this experiment is described in full.

4.3.1 Apparatus

The experiment ran on a standard workstation running Windows XP. The computer was a 3.2GHz PentiumD desktop machine with a 19" CRT monitor set to a resolution of 1280x1024 pixels, 32-bit colour. A standard three-button optical mouse was used, with default control-display gain and acceleration behaviour. The keyboard, also of a standard type, was not required for any performance related tasks.

The dataset visualised for this experiment was obtained from the Geography Department of the University of Canterbury, which provided a database containing the locations and details of traffic accidents that occurred in Christchurch over the 20-year interval of 1980 to 1999. Each crash record contained numerous descriptive fields, but for this experiment the important fields were Date and Location (expressed as latitude and longitude). The data was loaded from ESRI shape files using the Shapefile C Library (Warmerdam 2007). The complete crash database contained thousands of records and was reduced in two ways. Firstly, co-located crashes were removed, and secondly, crashes were partitioned into four groups, which were then cropped to match the chosen density levels for the experiment. These two reduction steps are expanded on below:

- A crash that occurred at approximately the same latitude and longitude as a previously stored crash was discarded because overlapping data items would complicate task completion. Partial occlusion was deemed acceptable up to a limit of half the radius of a data point as drawn on the screen.

- The data was partitioned into four groups based on the year of each crash. Each group spanned a five year interval: 1980-1984, 1985-1989, 1990-1994 and 1995-1999. At this point each group contained a different number of crashes, so crashes were randomly removed until all groups contained 200 crashes, which was the chosen size for the high density conditions. This process was repeated to produce the low density data set which contained 100 crashes per group.

Each of the four groups was the input for a data layer that could be visualised within the test application. In the terminology introduced earlier in this thesis, each group was a Model and each interface being evaluated provided a different transformation of the Model into a View.

4.3.2 Interfaces

An interface was designed for each visualisation technique: Static View, Global Filtering and Magic Lens. The interfaces were developed in C++

using Open Scene Graph (Osfield 2007). Each interface followed a similar layout in which a single application window ran in full-screen mode (1280x1024 pixels). The majority of this window was taken up by a street map that provided constant context throughout the tasks. Each crash data point was represented on the map as a filled circle with a radius of 8 pixels and a black outline. The circles for each of the four data groups were filled in a different colour. The colours were pastel shades of pink, green, blue and grey. At the top of the screen was a banner where the instructions for the current task were displayed. Neither zooming nor panning were required for any of the tasks so these functions were intentionally not provided.

The three visualisation techniques each present challenges for the user's perceptual abilities. A static view with no filtering increases the likelihood of information overload. Global filtering through selectable layers reduces this complexity by allowing the user to hide information that is not necessary for the current task. A Magic Lens also provides filtering. It reduces complexity by either restricting information to a bounded area or simplifying the view within that area. Refer to Figure 4.4 for screenshots. In particular, Figure 4.4(d) highlights the key differences between the three interfaces.

The Static View interface shown in Figure 4.4(a) is the control condition. It provides no means of managing the layers of information. Instead, all layers are overlaid and presented simultaneously.

The Global Filtering interface shown in Figure 4.4(b) uses a row of checkboxes to control the visibility of layers. A layer can be hidden when not required for a task, reducing the overall complexity of the view. When all layers are visible, this interface is effectively identical to the Static View interface.

The Magic Lens interface shown in Figure 4.4(c) is a combination of the Static View and Global Filtering interfaces. In Section 2.2, a taxonomy of views for supporting Focus and Context was presented. In reference to that taxonomy, the Magic Lens interface can be considered a *Composite View* with a *Spatial Separation*. As Figure 4.3 illustrates, the Magic Lens interface

is composed of the Static View with the Global Filtering View embedded within.

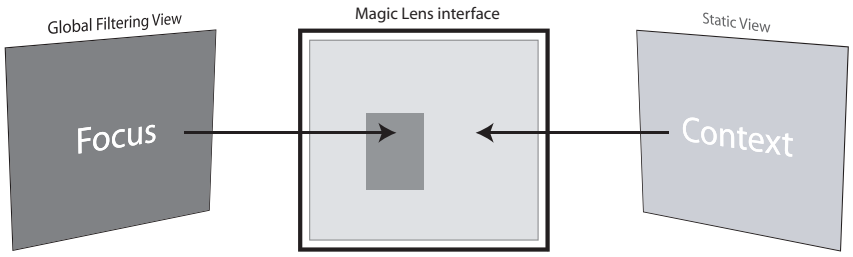


Figure 4.3: The Magic Lens interface is a Composite View with a Spatial Separation. It is the combination of the Static View (as the primary view) and the Global Filtering View (as the embedded secondary view).

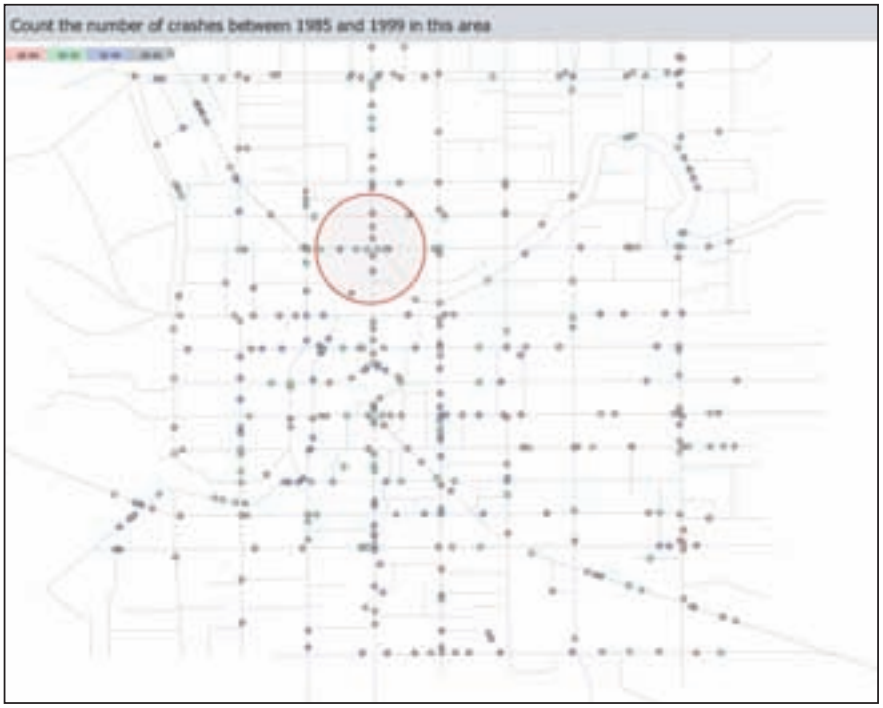
The Magic Lens interface allows the use of a single large workspace, while providing localised filtering. Within the lens area only those crash groups activated with the checkboxes are shown. Outside the lens area all crashes are shown as context.

The lens object itself was made up of several components. Attached to the top of the lens was the same set of checkboxes as the Global Filtering interface. The rest of the lens was a display area showing the filtered content. The display area was slightly tinted blue to distinguish it from the rest of the workspace. By default, the lens was 240x240 pixels in size and was positioned at the centre of the screen. A small rectangle anchored to the bottom right corner of the lens area could be dragged to resize the lens. Dragging any other part of the lens would move the lens around the workspace. The size and position of the lens was reset at the beginning of each task.

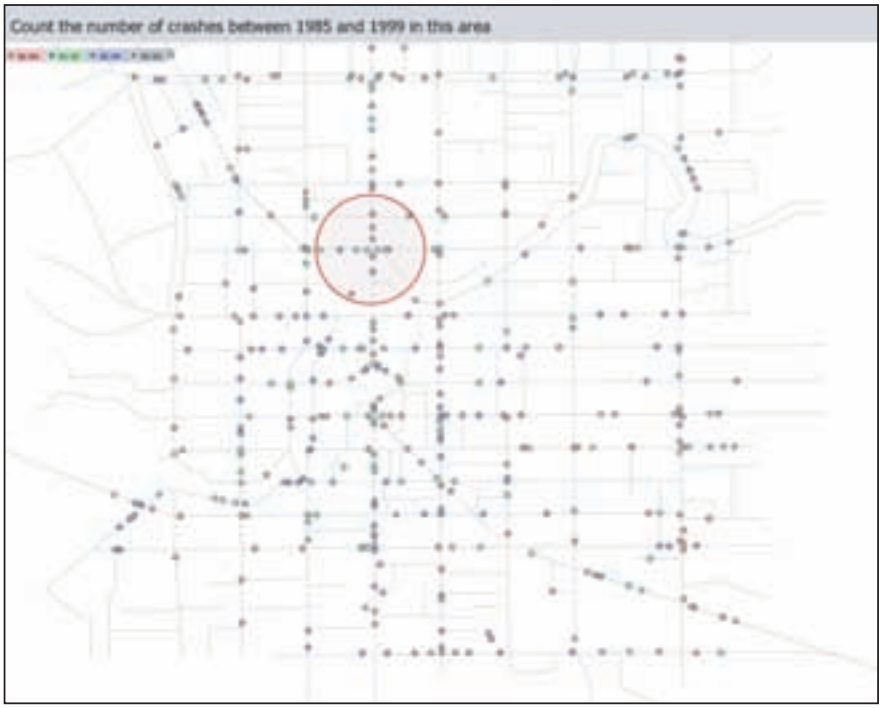
4.3.3 Participants

Sixteen participants (13 males and 3 females) were recruited from the Computer Science department. All participants had substantial experience with computers and were comfortable with the apparatus involved. Participants ranged in age from 22 to 41 and all were right handed.

Before starting the experiment, participants were tested to ensure they

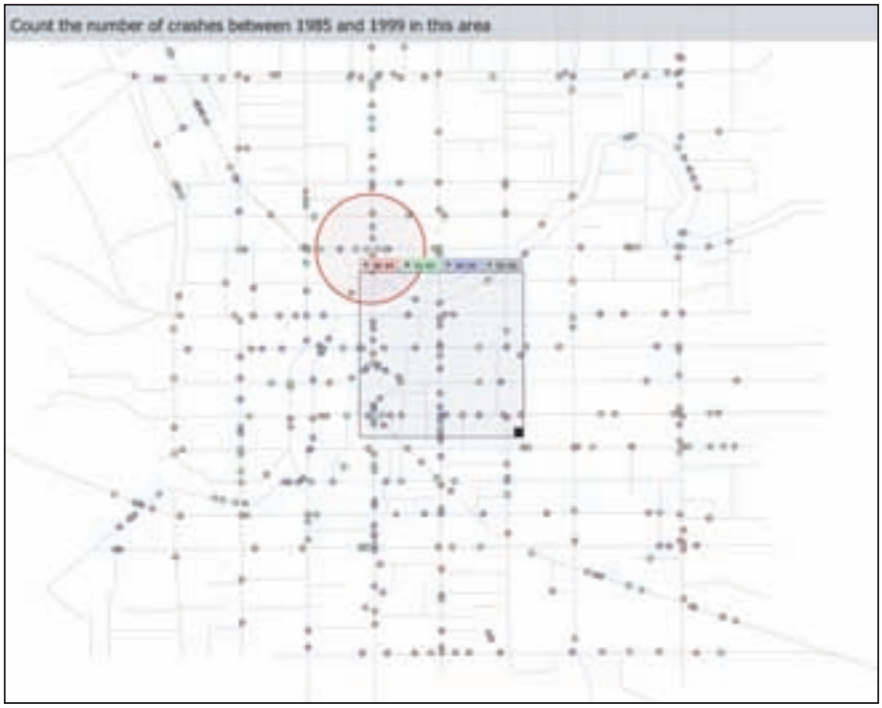


(a) The Static View interface provides no filtering.



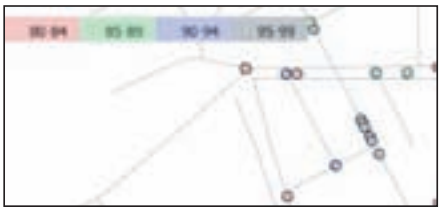
(b) The Global Filtering interface provided filtering through a row of checkboxes.

Figure 4.4: An overview of the three interfaces being evaluated.



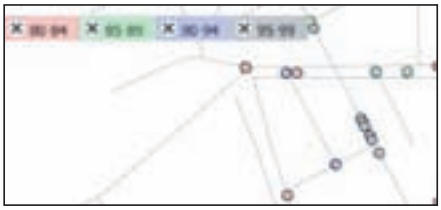
(c) The Magic Lens interface combines the Static View with filtering. Filtering only occurs within the lens area and is controlled with checkboxes attached to the lens.

Static View



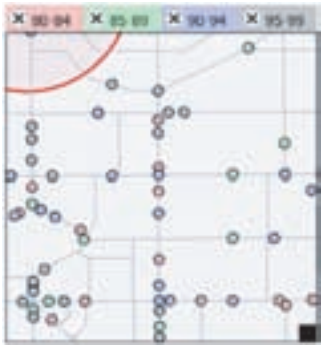
With this interface all layers are constantly visible.

Global Filtering



With this interface each layer can be shown or hidden using the checkboxes.

Magic Lens (Local Filtering)



With this interface each layer can be shown or hidden using the checkboxes, but the filter only applies within the lens area.

(d) A visual summary of the three interfaces being evaluated.

Figure 4.4: An overview of the three interfaces being evaluated (cont).

could easily distinguish between the four different colours used in the experiment. Each participant was shown a random array of coloured dots on the test screen and asked to name the four different colours. None of the participants had any trouble with this test and none mentioned having difficulties during the actual experiment.

4.3.4 Procedure

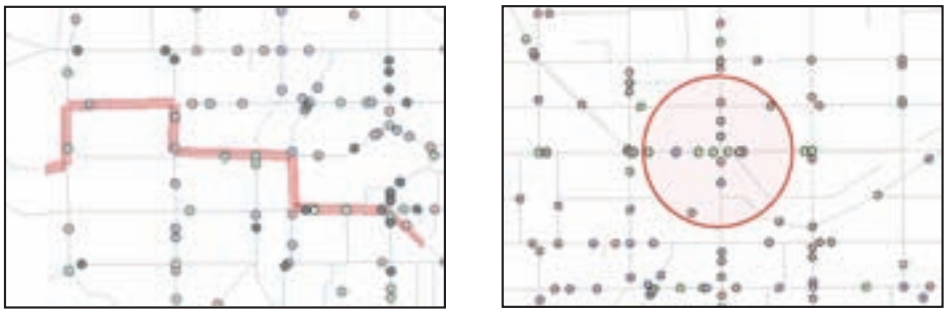
The experiment followed a 3x2x2x4 repeated measures design. The four factors were Interface, Density, Task Type and Layers. Interface had three levels (Static View, Global Filtering and Magic Lens), Density had two levels (Low and High), and Task Type had two levels (Path and Area). Layers was the number of data layers required to solve a task (1, 2, 3 or 4).

Density was measured as the number of crashes present on the map. The *High* density condition contained 800 items and the *Low* density condition contained 400 items. These were evenly distributed between the data layers (200 in each for high density, 100 in each for low density).

The participant was presented with a task statement displayed in a label at the top of the window. If they thought it necessary, the participant could then adjust the interface (if it supported adjustment) so that irrelevant data points were filtered from view. They would then count up the number of points that matched the task statement. Once satisfied with their count, they would hit the space-bar, the screen would go blank, and a text entry box would appear for them to type in their answer. Once they hit Enter to confirm their answer, the screen would return to the map and the next task would begin. All user actions were logged to files so that objective measures of performance could be calculated.

There were two different types of task presented to the participants. *Path* tasks displayed a highlighted route along a set of roads on the street map. The task was to count the number of crashes occurring along that route (see Figure 4.5(a)). An *Area* task displayed a shaded circular region on the map and asked the participant to count the number of crashes within that region (see Figure 4.5(b)). The participant was instructed to include in their count crashes touching the border of the path or area, and to remain aware that

some crashes might partially occlude others. They were asked to complete the tasks as quickly and accurately as possible.



(a) Path task: "Count the number of crashes along this path...".

(b) Area task: "Count the number of crashes in this area...".

Figure 4.5: Examples of the two task types.

Each task included a query component that required the participant to ignore 0, 1, 2 or 3 of the four groups of crashes. For example, if the task involved only crashes from 1980 until 1984, then the other three groups were not required and only served to clutter the interface. In the Static View interface, participants had to keep the query in mind as they counted because the interface provided no filtering aid. In the other interfaces, the participant could use the checkboxes to hide the groups of crashes that were not required.

After initial testing, an additional hint was added to the instructions to remind participants that only some crash groups might be required for the current task and that filtering was an option. The number of crash groups relating to the task were appended to the instruction. Three of the task instructions used in the experiment are shown here as examples:

- *Count the number of crashes between 1985 and 1999 in this area. (3 layers)*

- *Count the number of crashes between 1980 and 1984 along this path. (1 layer)*

- *Count the number of crashes between 1980 and 1999 in this area. (4 layers)*

The experiment was presented to participants as a set of six task blocks, representing the 3x2 combinations of Interface and Density. Participants used each interface with low density first and then with high density. The participants completed a practice round before using an interface type for the first time. They then completed the block of tasks and moved on to the next condition. Each task block consisted of 16 search tasks. There were two tasks for each combination of Task Type and Number of Layers. Figure 4.6 illustrates the structure of a task block.

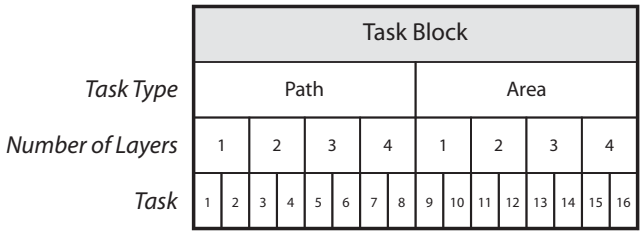


Figure 4.6: The structure of a task block. Of sixteen tasks, eight are Paths and eight are Areas. Of each eight, there are two tasks for each additional Layer.

The same task block could not be presented to participants in more than one condition because learning effects could bias the results. To resolve this problem, three equivalent task blocks were generated for low density and three for high density. These were randomly assigned to the conditions when a participant started the experiment. This meant that although all participants completed the same total 96 tasks, they did so in randomly different interfaces. In addition, tasks within a task block were presented in a random order.

Dependent variables for this experiment were Task Completion Time and Accuracy. Task Completion Time was measured as the interval between when the subject pressed enter to reveal the next task instructions, and when the subject pressed the space bar to indicate they had completed the

task to their satisfaction. Accuracy was measured as the fraction of task questions the user answered correctly. Based on these dependent measures, the following hypotheses were declared:

- H1: The Magic Lens interface is faster than Global Filtering and Static View.

- H2: The Magic Lens interface is more accurate than Global Filtering and Static View.

Questionnaires were used to collect subjective information about the interfaces. After each condition, subjects answered questions about their experience. In addition to specific questions related to the task, extra questions were adapted from the NASA TLX questionnaire (Hart and Staveland 1988) and the System Usability Scale (Brooke 1996). The exact questions posed are listed in Table 4.4 on page 121. They were presented as Likert scale responses, ranging from 1=Disagree and 7=Agree. The questionnaires are reproduced in full in Appendix A.

After using all three interfaces with low density, participants ranked the interfaces based on preference. Once all conditions had been completed, participants filled out a final questionnaire in which they again ranked the interfaces, this time based on overall preference. They were asked to provide any particular reasons they liked or disliked an interface, and to provide any overall observations.

4.4 Results

In this section the results from the experiment are presented. All statistical analysis was performed using SPSS version 15.0.

Global Filtering was the fastest interface. The fastest interface was Global Filtering (mean = 12797.357 ms, se = 631.475 ms), followed by Static View (mean = 14542.605 ms, se = 700.086 ms) and Magic Lens (mean = 14881.060 ms, se = 784.511 ms), producing a significant main effect of Interface on Task Completion Time ($F_{2,30} = 11.173$, $p < .05$). This result is

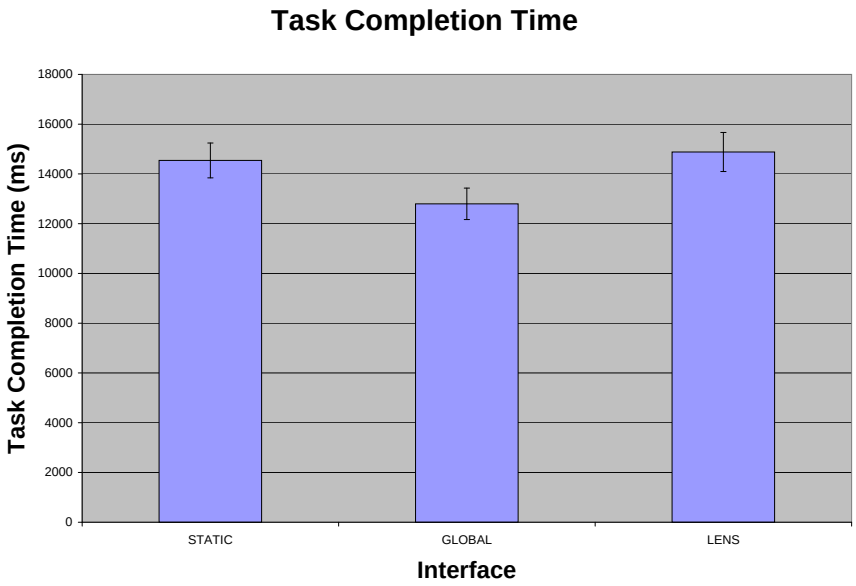


Figure 4.7: Task Completion Time.

shown in Figure 4.7. Post-hoc analysis of this effect using Bonferroni correction ($p < .05$) showed pairwise differences between Global Filtering and Static View, and Global Filtering and Magic Lens. This result means that hypothesis H1 cannot be accepted. The Magic Lens interface was not faster than Global Filtering or Static View.

Tasks were completed fastest in Low density. Tasks in the Low density conditions (mean = 10924.525 ms, se = 529.838 ms) were completed faster than tasks in High density (mean = 17222.823 ms, se = 835.068 ms) giving a significant main effect of Density on Task Completion Time ($F_{1,15}$ = 159.448, $p < .05$).

Tasks involving a single layer were completed fastest. Tasks requiring only one layer (mean = 10842.967 ms, se = 539.600ms) were completed fastest, followed by two layers (mean = 14181.104 ms, se = 644.506 ms), then four layers (mean = 15341.790 ms , se = 787.711 ms) and finally three layers (mean = 15928.834 ms, se = 767.819 ms). These values showed a significant

main effect of Number of Layers on Task Completion Time ($F_{3,45} = 74.071$, $p < .05$). Post-hoc analysis with Bonferroni correction ($p < .05$) showed pairwise differences between all levels except Three Layers and Four Layers, which was not significant.

Task Type (Area or Path) did not have a significant main effect on Task Completion Time.

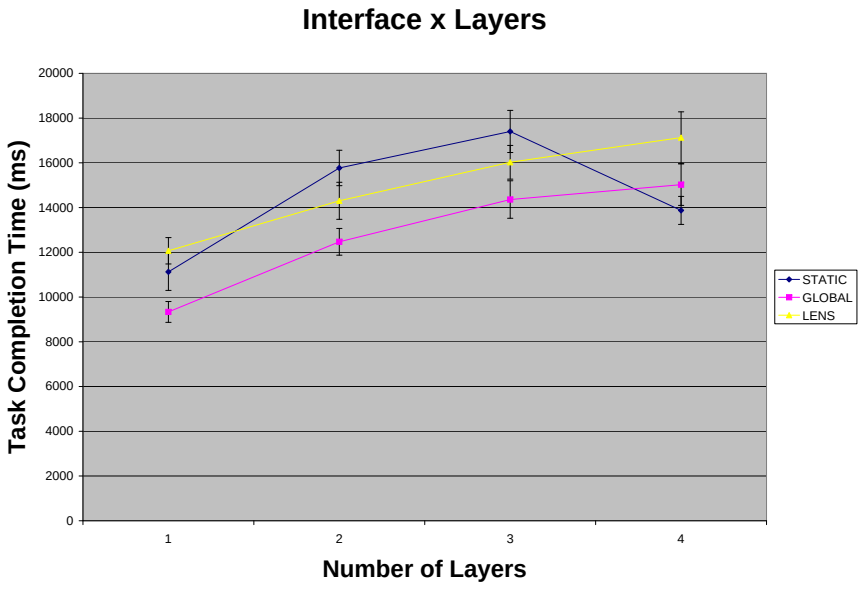


Figure 4.8: The Static interface improved in performance between Three and Four layers, while the other two interfaces decreased in performance.

The Static interface exhibits an interesting property. There was a significant interaction between Interface and Number of Layers ($F_{6,90} = 9.367$, $p < .05$). As Figure 4.8 shows, there was a sizable decrease in Task Completion Time between Three Layers and Four Layers for the Static interface, while there were increases for both the other interfaces. In fact, the Static interface is the fastest interface for tasks involving all four layers of information (mean = 13872.754 ms, se = 625.726). One explanation for this is that the Static interface did not permit any data filtering and the tasks with four layers did not require any. In comparison, tasks involving three

layers require the participant to ignore points of one particular colour when counting, which is a difficult cognitive activity.

The Magic Lens interface favoured Area tasks. There was a significant interaction between Interface and Task Type ($F_{2,30} = 18.207$, $p < .05$). Path tasks were completed faster than Area tasks in both the Static and Global Filtering interfaces. However, the trend is reversed for the Magic Lens interface, where Area tasks were completed significantly faster than Path tasks ($F_{1,15} = 11.201$, $p < .05$). This interaction is shown in Figure 4.9. This behaviour is likely due to the manipulation cost of resizing and moving the lens. An Area in this evaluation fit within the lens region without requiring resizing. Paths, on the other hand, were often long, requiring the user to either drag out the lens to cover the entire path, or drag the lens along the path as they counted. Either way, the task took longer. Solutions to this possible manipulation cost are explored in Section 4.5.

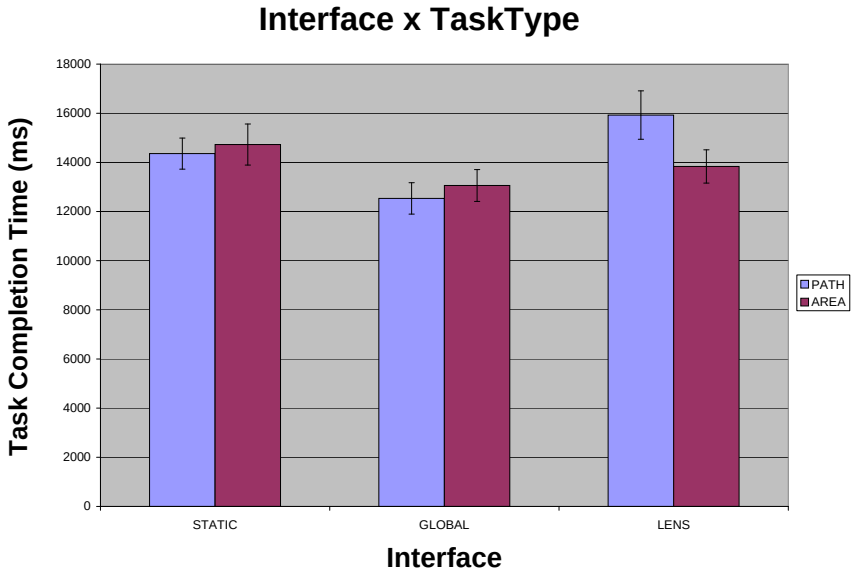


Figure 4.9: Path tasks were completed faster than Area tasks in the Static and Global Filtering interfaces, but the trend is reversed for the Magic Lens interface.

The following interactions were significant but were not interpreted because they do not involve Interface. They are included here for completeness.

- There was a significant interaction between Task Type and Number of Layers ($F_{3,45} = 21.649$, $p < .05$).
- There was a significant interaction between Density and Number of Layers ($F_{3,45} = 30.505$, $p < .05$).
- There was a significant interaction between Density, Task Type and Number of Layers ($F_{3,45} = 41.413$, $p < .05$).

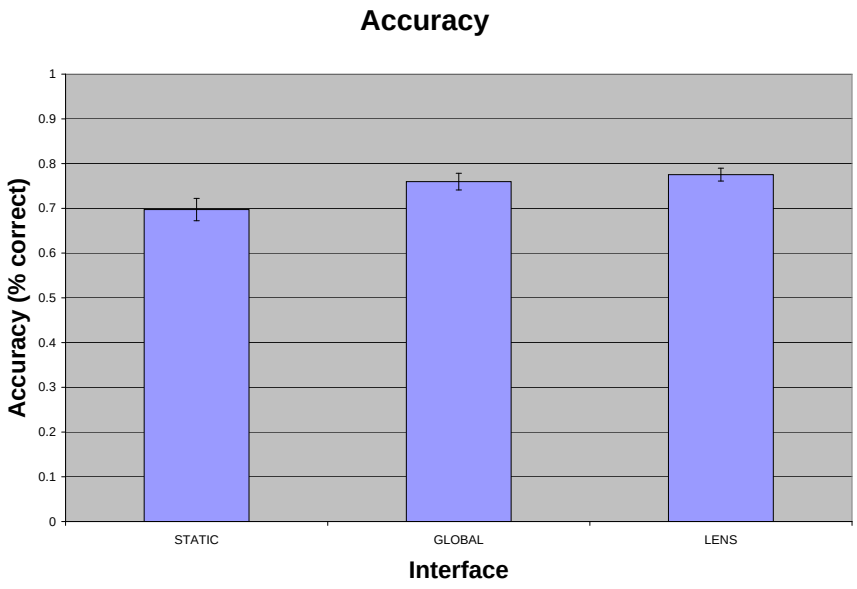


Figure 4.10: Accuracy.

The Magic Lens interface had high accuracy. The interface with highest recorded accuracy was the Magic Lens (mean = 0.775, se = 0.014) followed by Global Filtering (mean = 0.760, se = 0.019) and Static View (mean = 0.697, se = 0.025), producing a significant main effect of Interface on Accuracy ($F_{2,30} = 5.362$, $p < .05$). This result is shown in Figure 4.10. Post-hoc

analysis with Bonferroni correction ($p < .05$) showed pairwise differences between Static View and the other two interfaces, but none between Global Filtering and Magic Lens. This result means that hypothesis H2 can be conditionally accepted: that Magic Lenses were more accurate than the other two interfaces, although the difference with Global Filtering was not statistically significant.

Accuracy was higher in low density. Tasks in Low density were completed more accurately (mean = 0.861, se = 0.015) than those in High Density (mean = 0.628, se = 0.017), producing a main effect of Density on Accuracy ($F_{1,15} = 197.054$, $p < .05$).

Tasks involving a single layer were completed the most accurately. Tasks requiring only one layer were completed with the highest accuracy (mean = 0.885, se = 0.013), followed by two layers (mean = 0.727, se = 0.016), three layers (mean = 0.724. se = 0.024) and four layers (mean = 0.641, se = 0.030). These values showed a significant main effect of Number of Layers on Accuracy ($F_{3,45} = 26.961$, $p < .05$). Post-hoc analysis with Bonferroni correction ($p < .05$) showed pairwise differences between One Layer and all other levels, but no other differences were significant.

Task Type (Area or Path) did not have a significant main effect on Accuracy.

The following interactions were significant but were not interpreted because they do not involve Interface. They are included here for completeness.

- There was a significant interaction between Density and Task Type ($F_{1,15} = 8.692$, $p < .05$).

- There was a significant interaction between Density and Number of Layers ($F_{3,45} = 6.259$, $p < .05$).

4.4.1 Subjective Results

Participants had minimal trouble understanding the interfaces. The means of responses to the question "I found the interface easy to use"

were over six out of seven for all three interfaces. Global Filtering was rated highest (mean = 6.594, se = 0.131), followed by Magic Lens (mean = 6.500, se = 0.171) and Static View (mean = 6.125, se = 0.355). These values did not produce a main effect.

The Static View interface was the least easy to use. When asked if they found the tasks easy to complete with the interface, participants rated Magic Lens the highest (mean = 5.625, se = 0.207), followed by Global Filtering (mean = 5.344, se = 0.222) and finally Static View (mean = 3.250, se = 0.326), producing a significant main effect ($F_{2,30} = 38.585$, $p < .05$). Post-hoc analysis with Bonferroni correction ($p < .05$) showed pairwise differences between Static View and other two interfaces, but no difference between Global Filtering and Magic Lens. Tasks were considered easier in Low density (mean = 5.313, se = 0.225) than in High density (mean = 4.167, se = 0.234), producing a significant main effect ($F_{1,15} = 21.174$, $p < .05$).

Participants felt fast with the Magic Lens. Participants felt they performed quickly in the Magic Lens interface (mean = 5.094, se = 0.220), and the Global Filtering (mean = 4.906, se = 0.200) and less so in Static View (mean = 3.188, se = 0.332), providing a significant main effect ($F_{2,30} = 20.906$, $p < .05$). Post-hoc analysis with Bonferroni correction ($p < .05$) showed pairwise differences between Static View and the other two interfaces, but no difference between Global Filtering and Magic Lens. Participants felt they performed quickest in Low density (mean = 4.958, se = 0.202) rather than High density (mean = 3.833, se = 0.202), producing a significant main effect ($F_{1,15} = 32.642$, $p < .05$).

Participants felt confident with the Magic Lens. Participants were confident about their accuracy in the Magic Lens interface (mean = 5.125, se = 0.217) and the Global Filtering (mean = 4.813, se = 0.209) and less so in Static View (mean = 3.250, se = 0.285) producing a significant main effect ($F_{2,30} = 27.613$, $p < .05$). Post-hoc analysis with Bonferroni correction ($p < .05$) showed pairwise differences between Static View and the other two interfaces, but no difference between Global Filtering and Magic Lens.

Confidence was higher in Low density (mean = 4.854, se = 3.937) compared to High density (mean = 3.937, se = 0.226) producing a significant main effect ($F_{1,15} = 15.000$, $p < .05$).

Participants liked the Magic Lens for the tasks. Participants were asked whether they would appreciate using an interface if they were required to perform these sort of tasks on a regular basis. Participants reported the highest rating for the Magic Lens interface (mean = 5.500, se = 0.270) followed by Global Filtering (mean = 4.938, se = 0.322) and lastly Static View (mean = 2.094, se = 0.314). Post-hoc analysis with Bonferroni correction (p $< .05$) showed pairwise differences between Static View and the other two interfaces, but no difference between Global Filtering and Magic Lens.

The Static View and High Densities were physically demanding. Static View was rated the most physically demanding (mean = 3.594, se = 0.529), followed by Global Filtering (mean = 3.031, se = 0.404) and then Magic Lens (mean = 2.938, se = 0.356), although the effect was not significant. However, participants found the High density conditions to be more physically demanding (mean = 3.521, se = 0.479) than the Low density conditions (mean = 2.854, se = 0.363), producing a significant main effect ($F_{1,15}$ = 7.619, $p < .05$).

The Magic Lens was least mentally demanding. Participants found the Static View interface the most mentally demanding (mean = 5.906, se = 0.307), followed by Global Filtering (mean = 4.563, se = 0.395) and then Magic Lens (mean = 3.813, se = 0.362), producing a significant main effect ($F_{2,30} = 32.340$, $p < .05$). Post-hoc analysis with Bonferroni correction (p $< .05$) showed pairwise differences between all three interfaces. The effect of Density on mental demand was not significant.

The Static View was frustrating. Static View was found to be the most frustrating interface (mean = 5.188, se = 0.338), followed by Global Filtering (mean = 3.281, se = 0.362) and then Magic Lens (mean = 2.594, se = 0.247), producing a significant main effect ($F_{2,30} = 24.265$, $p < .05$). Post-

hoc analysis with Bonferroni correction ($p < .05$) showed pairwise differences between Static View and the other two interfaces, but no difference between Global Filtering and Magic Lens. Density was also significant with High density being more frustrating than Low density ($F_{1,15} = 12.181$, $p < .05$).

Preferences

The Magic Lens interface was clearly the interface of choice in this experiment. It was most preferred in Low Density (Friedman Test $\chi^2_r = 24.1$, df=2, N=16, $p < .05$), High Density (Friedman Test $\chi^2_r = 26.0$, df=2, N=16, $p < .05$), and overall (Friedman Test $\chi^2_r = 26.8$, df=2, N=16, $p < .05$).

4.5 Additional Analysis of Manipulation Cost

The results of the experiment found the Magic Lens to be the slowest performing interface. The Magic Lens attempts to merge the contextual support of Static View with the focusing capabilities of Global Filtering. Why then did the Magic Lens interface perform poorly? From observations of the participants taking part in the experiment and comments from questionnaires, it is suspected that the Magic Lens introduced high manipulation costs into the interface. Therefore, mouse input data was analysed to test the following hypothesis:

- H3: The Magic Lens incurs a higher Manipulation Cost than Static View or Global Filtering.

In the experiment participants used the mouse to control the interfaces. Therefore, Manipulation Cost was quantified according to the number and duration of mouse input events. The dependent variables were Movement Distance, Drag Distance and Number of Clicks. These measures were computed from data extracted from log files made during the experiment. All mouse movements and button actions had been recorded.

Movement Distance, measured in pixels, was the average distance the mouse cursor travelled during a block of tasks. Drag Distance, also measured in pixels, was the amount of Movement Distance that occurred while the

mouse button was down. Number of Clicks was simply the average number of times the mouse button was clicked during a block of tasks.

The analysis of Manipulation Cost followed the same 3x2x2x4 experimental design as the analyses of Task Completion Time and Accuracy reported previously.

4.5.1 Results

A summary of Movement Distance, Drag Distance and Number of Clicks for each condition is provided in Table 4.5.

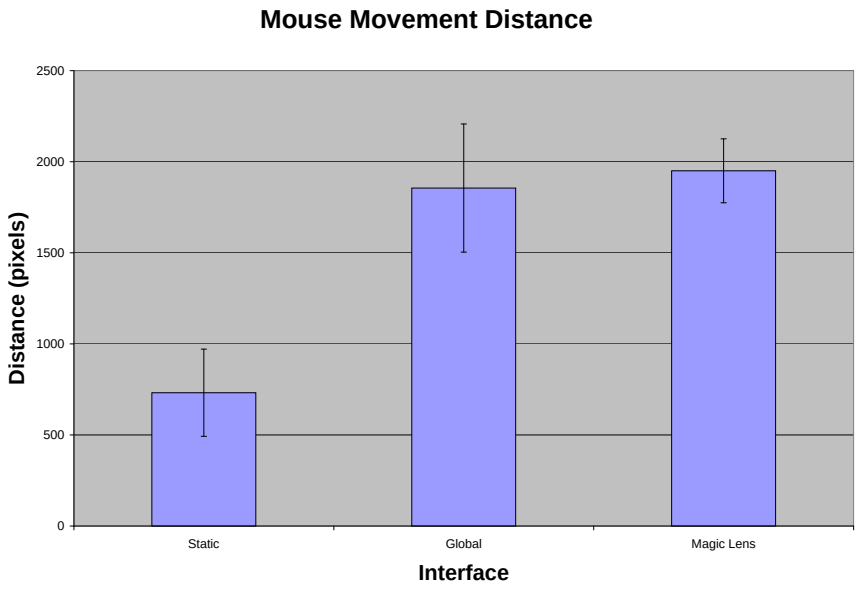


Figure 4.11: Mouse Movement.

Movement Distance had a grand mean of 1512.6 pixels (se = 243.1 pixels). As a point of reference, the display used in this experiment had a resolution of 1280x1024 pixels, so the average movement distance for each task was approximately the same as the diagonal distance from the top left to the bottom right corners of the screen.

The mouse was moved least in the Static View interface. Static View saw the least amount of mouse movement (mean = 731.9 pixels, se =

239.4 pixels), followed by Global Filtering (mean = 1855.6 pixels, se = 351.9 pixels) and Magic Lens (mean = 1950.3 pixels, se = 175.5 pixels). These values, plotted in Figure 4.11, produced a significant main effect ($F_{2,30}$ = 26.509, $p < .05$). Post-hoc analysis with Bonferroni correction (p < .05) showed pairwise differences between Static View and the other two interfaces, but no difference between Global Filtering and Magic Lens.

There was less mouse movement in Low density. Mouse movement was lower in Low density tasks (mean = 1303.9 pixels, se = 199.2 pixels) than in High density tasks (mean = 1721.3 pixels, se = 291.0 pixels), producing a significant main effect ($F_{1,15} = 14.141$, $p < .05$).

There was more mouse movement for Path tasks. Path tasks required more mouse movement (mean = 1581.8 pixels, se = 230.2 pixels) than Area tasks (mean = 1443.4 pixels, se = 258.8 pixels), giving a significant main effect ($F_{1,15} = 5.469$, $p < .05$).

The number of layers required for tasks had no significant effect on mouse movement.

Table 4.1 provides a list of interactions between factors that were significant for Movement Distance. These interactions were not further analysed, but are included here for completeness.

Interface x TaskType	$F_{2,30} = 50.140$, $p < .05$
Interface x Density x TaskType	$F_{2,30} = 3.816$, $p < .05$
Interface x Density x Layers	$F_{6,90} = 2.764$, $p < .05$
Density x Layers	$F_{3,45} = 5.068$, $p < .05$
Density x TaskType x Layers	$F_{3,45} = 3.768$, $p < .05$
TaskType x Layers	$F_{3,45} = 4.384$, $p < .05$

Table 4.1: Additional significant interactions between factors for Movement Distance.

Dragging occurred more in the Magic Lens interface. Participants dragged substantially further in the Magic Lens interface (mean = 424.8 pixels, se = 24.294 pixels) than in either of the other two interfaces, in

which dragging was non-existent or negligible (mean < 1 pixel, se < 1 pixel). These differences produced a significant main effect ($F_{2,30} = 305.433$, $p < .05$). Post-hoc analysis with Bonferroni correction (p < .05) showed pairwise differences between Magic Lens and the other two interfaces, but no difference between Global Filtering and Static View.

Dragging occurred more in High density. There was more dragging activity in High Density (mean = 154.512 pixels, se = 6.907 pixels) than in Low Density (mean = 129.299 pixels, se = 10.381 pixels), producing a significant main effect ($F_{1,15} = 13.223$, $p < .05$).

Path tasks involved more dragging than Area tasks. There was a also significant main effect of Task Type on Drag Distance ($F_{1,15} = 87.715$, $p < .05$). Path tasks involved more dragging (mean = 168.992 pixels, se = 10.208 pixels) than Area tasks (mean = 114.819 pixels, se = 6.630 pixels).

Table 4.2 provides a list of interactions between factors that were significant for Drag Distance. These interactions were not further analysed, but are included here for completeness.

Interface x Density	$F_{2,30} = 13.100$, $p < .05$
Interface x Layers	$F_{6,90} = 2.658$, $p < .05$
Interface x TaskType	$F_{2,30} = 89.473$, $p < .05$
Interface x TaskType x Layers	$F_{6,90} = 7.584$, $p < .05$
Interface x Density x TaskType	$F_{2,30} = 17.301$, $p < .05$
Interface x Density x TaskType x Layers	$F_{6,90} = 6.307$, $p < .05$
Density x TaskType	$F_{1,15} = 16.991$, $p < .05$
Density x TaskType x Layers	$F_{3,45} = 6.233$, $p < .05$
TaskType x Layers	$F_{3,45} = 7.643$, $p < .05$

Table 4.2: Additional significant interactions between factors for Drag Distance.

The Magic Lens interface required more clicking. The Magic Lens interface saw the most mouse clicks (mean = 3.504, se = 0.288), followed by Global Filtering (mean = 2.033, se = 0.256) and then Static View with

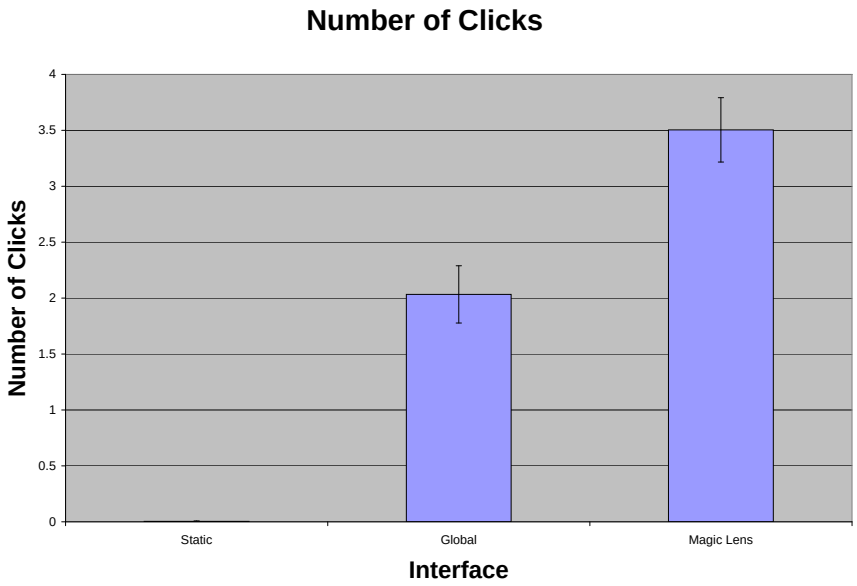


Figure 4.12: Number of Clicks.

essentially no clicking at all (mean = 0.004, se = 0.004). These results produced a significant main effect ($F_{2,30} = 113.243$, $p < .05$). Post-hoc analysis with Bonferroni correction ($p < .05$) showed that all interfaces were significantly different. See Figure 4.12.

There was more clicking in High density tasks. Tasks in High density required more clicks (mean = 2.133 se = 0.241) than Low density (mean = 1.561, se = 0.137), producing a significant main effect ($F_{1,15} = 11.427$, $p < .05$).

There was more clicking for Path tasks. Path tasks required more clicks (mean = 2.003 se = 0.184) than Area tasks (mean = 1.691 se = 0.177), giving a significant main effect ($F_{1,15} = 20.385$, $p < .05$).

There was more clicking in tasks involving a single layer. Tasks involving one layer required the most mouse clicks, producing a significant main effect($F_{3,45} = 8.384$, $p < .05$).

Table 4.3 provides a list of interactions between factors that were significant for Number of Clicks. These interactions were not further analysed, but are included here for completeness.

Interface x Density	$F_{2,30} = 8.805$, $p < .05$
Interface x TaskType	$F_{2,30} = 32.202$, $p < .05$
Interface x Layers	$F_{6,90} = 6.188$, $p < .05$
Density x Layers	$F_{3,45} = 5.067$, $p < .05$
Interface x Density x Layers	$F_{6,90} = 4.723$, $p < .05$

Table 4.3: Additional significant interactions between factors for Number of Clicks.

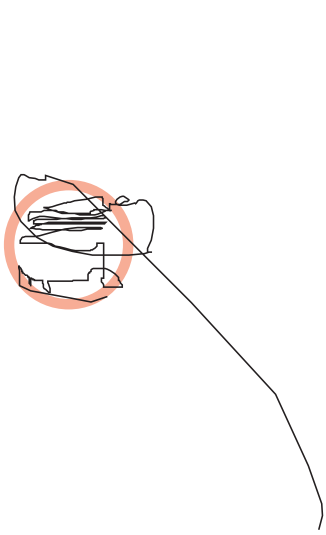
4.5.2 Visual Analysis of Mouse Movement

In addition to analysing the dependent variables Mouse Movement, Drag Ratio and Number of Clicks, the trajectories of participants' cursors were also plotted so that patterns of movement could be examined. Immediately one particular behaviour was obvious: participants used the mouse cursor to help them keep track of their counting. The paths clearly show concentrated mouse movements within the task Areas and along the task Paths (see Figure 4.13(a)). Also obvious are the long trips required in the Global Filtering interface to reach the checkboxes at the top left of the screen (see Figure 4.13(b)). Stone et al. (1994) claimed one of the benefits of ToolGlass and Magic Lenses was how they "bring tools to the data". The analysis of mouse movements provides visual evidence of this claimed benefit.

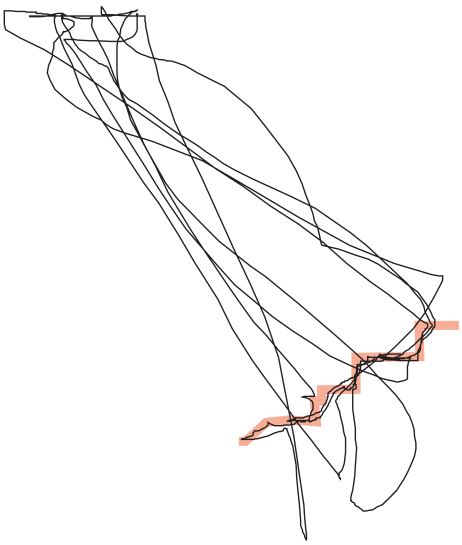
4.5.3 Summary of Manipulation Cost

- Magic Lenses required more mouse movement than the other techniques, even more than Global Filtering which required large movements to reach the checkboxes in the top left of the screen.

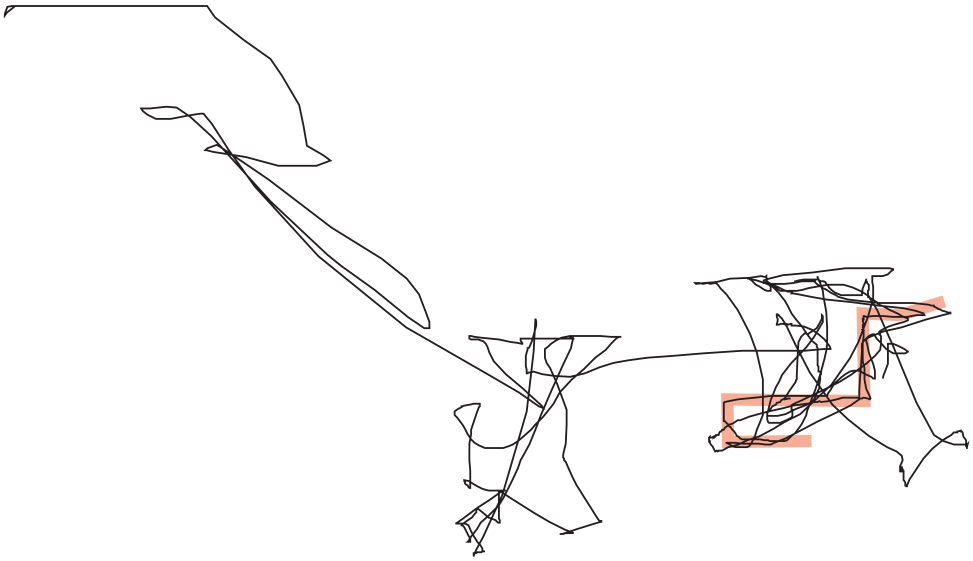
- Magic Lenses required a large amount of mouse dragging compared to no dragging at all in the other interfaces. Dragging was used to position



(a) A representative mouse path from a trial in the Static View Interface. Even though there are no checkboxes to click or a lens to drag, the participant still makes extensive use of the mouse. It is clear from the path that the mouse cursor is being used as a counting aid.



(b) A representative mouse path from a trial in the Global Filtering interface. Notice how many long mouse movements are required to visit the checkboxes in the top left corner.



(c) A representative mouse path from a trial in the Magic Lens interface. This path demonstrates a mistake where the participant reached for the top left corner where the checkboxes would be in the Global Filtering interface.

Figure 4.13: Visualisations of representative mouse paths from each interface.

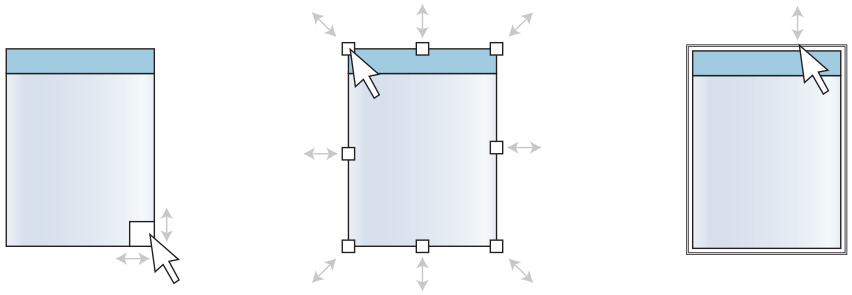
and resize the lens. Over 20% of mouse movement in the Magic Lens conditions was dragging.

- Magic Lenses required a greater number of mouse clicks to use, likely caused by the need for frequent dragging actions.

Overall it would appear that there were substantially more user actions required to use the Magic Lens interface. These are associated with the positioning and resizing that was necessary with the Magic Lens. Ways to address this problem are discussed in the following section.

4.5.4 Reducing the cost of Magic Lens Manipulation

The analysis of mouse actions in the three interfaces revealed a disproportionate amount of mousing activity in the Magic Lens interface. Here several alternative Magic Lens manipulation strategies are suggested that could reduce the large manipulation cost.



(a) The resize technique implemented in the experiment.

(b) A full "compass" of eight resize handles.

(c) The lens border can act as a single continuous resize handle.

Figure 4.14: Magic Lens resizing strategies.

The resizing mechanism in the Magic Lens implementation was rudimentary. A resizing handle in the bottom right corner of the lens could be used to drag the lens area larger or smaller (see Figure 4.14(a)).

The single resize handle meant that the lens had to be positioned in the top left corner of the target area and then resized to cover it. The lens could not be expanded in any direction other than south-east. This constraint was an obvious cause of frustration for participants. Typical usage of the lens involved the participant dragging it to the right position and then resizing. These two steps were distinct activities, and often the participant had already acquired the resize handle and begun resizing, only to find they had not positioned the lens correctly and would need to go back a step. One possible solution would be to arrange a full set of eight drag handles around the lens (shown in Figure 4.14(b)), allowing it to be resized in any direction.

A further refinement would be to turn the entire lens border into a continuous resize handle (shown in Figure 4.14(c)), in the same way as most desktop operating systems allow the user to resize windows. With window resizing being so ubiquitous - Gaylin (1986) claims 2% of all window commands involve resizing - a number of novel techniques have been researched in order to increase efficiency in this area, including Semantic Pointing (Blanch, Guiard and Beaudouin-Lafon 2004), in which the motor-space associated with a target (e.g. a window border) is increased while its visual representation remains the same.

Alternatively, a separate input could be used to control lens size, such as the scroll-wheel on the mouse. The scroll wheel is already an accepted means for zooming in many user interfaces. For example, holding down the control-key and rolling the scroll wheel adjusts the zoom level of the document in Microsoft Office applications.

The Magic Lenses Spraycan tool is a hypothetical interface concept leading directly from observations of participants in the evaluation, feedback from the questionnaires, and reflection on metaphors that could make the tasks presented in this experiment more intuitive and enjoyable. In this new interface, the rectangular Magic Lens is abandoned in favour of freeform region lenses. These are arbitrarily-shaped regions that behave like a rectangular Magic Lens - in that they filter the information they currently cover - but the method of application is based on a different metaphor: that of a spray can. The spray can metaphor is already established in painting applications,

where spray tools administer bold strokes of colour. In the Spraypaint Magic Lens interface, shown in Figure 4.15 on page 123, the colours painted by the spray can map directly to the datasets being investigated. The paint blocks out information items that don't match its colour. The colour mixing that occurs where regions touch is synonymous with having multiple overlapping Magic Lenses.

Reducing the need for Mouse Counting

The concept of Safe Exploration, introduced in Section 3.1, could be used to make the tasks in this experiment a great deal easier. Safe Exploration is a method of using the Magic Lens to explore potential changes to the underlying workspace without the risk of corrupting that workspace.

Participants showed a tendency to use the mouse cursor as a counting aid, in the same way as people use their finger to keep track of counting items in the real world. However, in the real world, people often use a shortcut to help them count: they remove items from the set as they are counted, leaving only those left to be counted behind.

This approach could be utilised in the interfaces of this experiment. As the participant counted an item, they could click on it to mark it, leaving only those left to be counted unmarked. Naturally, this approach could be applied in all three of the interfaces, however applying it within the Magic Lens creates a tidy separation between the original data and the current task. The Magic Lens can simply be discarded once counting is complete.

4.6 Summary

The first hypothesis, that the Magic Lens interface was faster than Static View and Global Filtering, was rejected. In fact, the Magic Lens interface performed slowest. The second hypothesis, that the Magic Lens was more accurate than the other two techniques, was conditionally accepted. The Magic Lens interface had the highest recorded accuracy, but post-hoc analysis found the accuracy difference between Magic Lens and Global Filtering was not statistically significant.

Subjective results were overwhelmingly positive in favour of the Magic

Lens interface. Participants claimed it caused the lowest frustration and mental demand, and they felt more confident and efficient with the lens. This result is interesting because the objective results show that performance was not higher with the lens and accuracy was not significantly so.

Participants appreciated the lens' ability to provide an *Explicit Area of Focus*, one of the stated advantages of Magic Lenses (see Section 3.1).

The slow performance of the Magic Lens prompted the additional investigation of mouse input in each of the interfaces. This analysis showed that when participants used the Magic Lens interface, they moved the mouse more, dragged the mouse more, and made more mouse clicks. This suggests that the cost of manipulating the Magic Lens was high, and that optimising this aspect of the lens may provide great benefits. Various approaches to address this problem were proposed, although their exploration and evaluation remain task for future work.

		Static View				Global Filtering				Magic Lens			
		Low		High		Low		High		Low		High	
Question		mean	se	mean	se	mean	se	mean	se	mean	se	mean	se
1.	I found the interface easy to understand	6.188	0.401	6.063	0.403	6.625	0.125	6.563	0.157	6.438	0.203	6.563	0.157
2.	I found the tasks easy to complete	3.938	0.461	2.563	0.353	5.934	0.266	4.750	0.266	6.063	0.213	5.188	0.306
3.	I feel that I performed quickly	3.688	0.416	2.688	0.384	5.500	0.274	4.313	0.270	5.688	0.254	4.500	0.258
4.	I feel confident about my accuracy	3.750	0.335	2.750	0.359	5.375	0.287	4.250	0.281	5.438	0.241	4.813	0.277
5.	If I had to carry out this sort of work on a regular basis, this is an interface I would appreciate using	2.500	0.408	1.688	0.285	5.313	0.362	4.563	0.387	5.563	0.316	5.438	0.302
6.	I found this condition physically demanding	3.313	0.489	3.875	0.576	2.625	0.407	3.438	0.508	2.625	0.364	3.250	0.403
7.	I found this condition mentally demanding	5.688	0.299	6.125	0.407	4.250	0.371	4.875	0.455	3.688	0.362	3.938	0.423
8.	I found this condition frustrating	4.813	0.485	5.563	0.302	2.688	0.384	3.875	0.386	2.188	0.332	3.000	0.258

Table 4.4: Summary of questionnaire responses. Questions were posed on a seven-point Likert scale between 1 = Disagree and 7 = Agree.

		Movement Distance (pixels)		Drag Distance (pixels)		Number of Clicks	
		mean	se	mean	se	mean	se
Low Density	Static	467.031	213.462	0.000	0.000	0.000	0.000
	Global	1628.207	281.160	0.629	0.145	1.719	0.197
	Magic Lens	1816.438	181.889	387.270	31.123	2.965	0.225
High Density	Static	996.793	305.562	0.000	0.000	0.008	0.008
	Global	2082.953	434.775	1.160	0.231	2.348	0.351
	Magic Lens	2084.156	198.516	462.375	20.650	4.043	0.400

Table 4.5: Overall statistics for the dependent variables of Manipulation Cost.

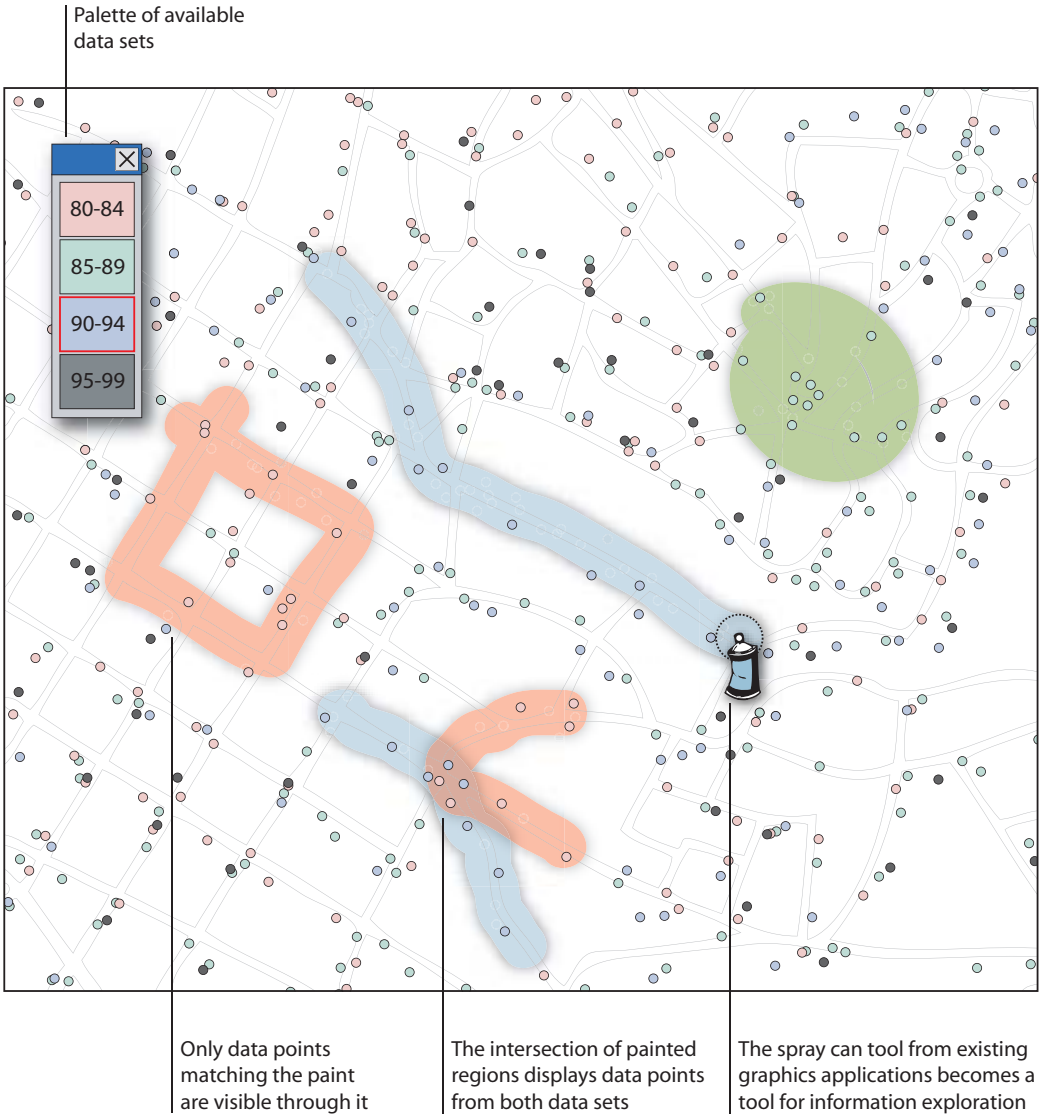


Figure 4.15: A potential variation of Magic Lenses in which filters are not restricted to rectangles, but are freeform regions created using the familiar spray can metaphor from graphics applications.

Chapter 5

Augmented Reality Magic Lenses

5.1 Introduction

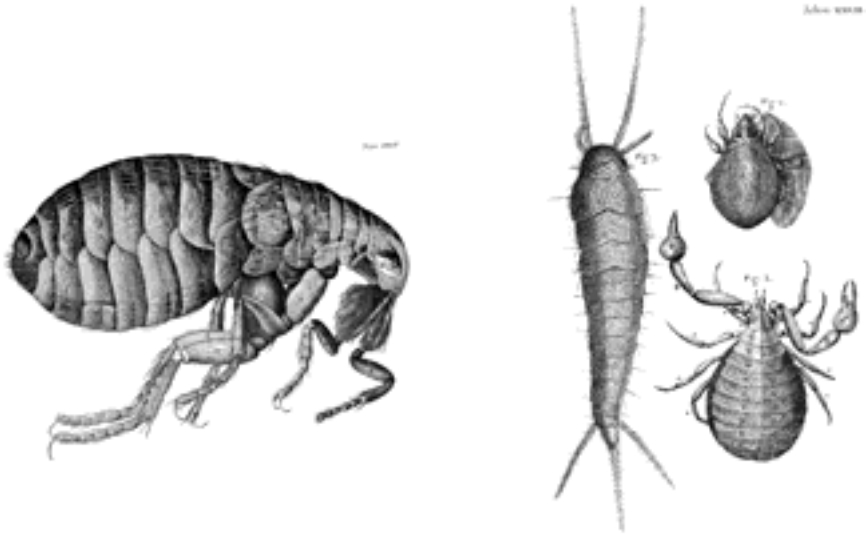
This chapter describes the development of Augmented Reality Magic Lenses: the extension of Magic Lenses into Tangible Augmented Reality user interfaces. Augmented Reality (AR) was described in detail in Section 2.4. Here we present the motivation for applying Magic Lenses in AR, a set of implementation techniques that can be used to achieve this, and a collection of trial applications that demonstrate various applications of AR Magic Lenses.

In developing AR Magic Lenses, we took an explorative approach and regularly conducted informal usability studies to obtain user feedback. AR Magic Lenses have been regularly shown in public open days at HIT Lab NZ and these occasions provided many interesting insights into Magic Lens use.

5.2 Motivation for AR Magic Lenses

AR is a technology that can enhance our view of the real world, while at the same time supporting tangible interaction through tracked tools. These two features each present opportunities for compelling new user interfaces that blur the boundary between the physical and digital worlds. Visual enhancement and Tangible Interaction are discussed below.

Visual enhancement of one's surroundings is a powerful ability. Sight is the most relied upon of the five human senses, although at certain limits our vision becomes inadequate. We cannot discern objects below a certain size or beyond a certain distance. In order to observe these phenomena in detail, people developed tools to enhance their vision (e.g. Figure 5.1), as identified by Robert Hooke:



(a) A flea. (b) A bookworm.

Figure 5.1: Drawings made by Robert Hooke after observations he made through magnifying lenses. Images from the public domain version of *Micrographia* available through Project Gutenberg.

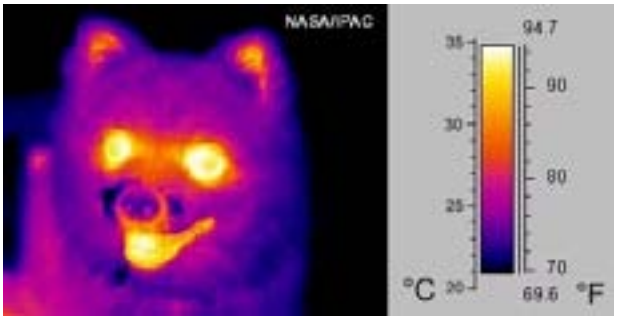
> The next care to be taken, in respect of the Senses, is a supplying of their infirmities with Instruments, and, as it were, the adding of artificial Organs to the natural; this in one of them has been of late years accomplisht with prodigious benefit to all sorts of useful knowledge, by the invention of Optical Glasses. By the means of Telescopes, there is nothing so far distant but may be represented to our view; and by the help of Microscopes, there is nothing so small, as to escape our inquiry; hence there is a new visible World discovered to the understanding.
>
> Robert Hooke, in *Micrographia* (1665).

While such tools provide increased optical clarity, much information remains hidden, sometimes because it is not within the colour spectrum, other times because it is not a physical property that can be detected and measured, such as abstract notions of ownership, worth, purpose or name.

Our vision permits us to see only a small portion of the electromagnetic spectrum. A number of technologies exist that can transform otherwise invisible wavelengths into observable forms. Examples include night vision equipment (Figure 5.2(a)), X-rays (Figure 5.2(b)) and thermal (infrared) imaging (Figure 5.2(c)). Although not strictly classed as AR interfaces, these tools demonstrate the scale of information that exists but is not seen.

(a) Night Vision. Image public domain.

(b) X-Ray Image. Image used under Creative Commons license.



(c) Thermal Imaging. Image public domain.

Figure 5.2: Visual representations of otherwise invisible phenomena.

In addition to their appearance, objects have many physical characteristics, such as a location in the world, dimensions and component parts. Furthermore, objects also have less tangible properties, such as a date of manufacture, a price, a country of origin and so on. In fact, every object within our field of view is associated with information of various kinds, both observable and non-observable, giving the net effect of an extremely dense, highly dimensional information space expanding in every direction. AR user

interfaces provide the opportunity to annotate our surroundings, visually revealing and managing the information hidden therein.

In an AR user interface where the physical surroundings are enhanced, re-represented and swimming with new information, techniques to manage increased visual information density will be vitally important. AR Magic Lenses are filtering tools that can be used in these interfaces to help us manage this complexity. AR Magic Lenses can be used to partition the user's view into areas of Focus and Context, separated by the lens border.

Tangible interaction allows the user to control the AR Magic Lens in an intuitive way, as they can apply their knowledge and experience of real magnifying glasses to their use of AR Magic Lenses. To reveal more information, the user can bring the lens closer to their eyes, thereby enlarging the focus area. In contrast, moving the lens farther away reduces the size of the focus area, and putting the AR Magic Lens away hides the focus area completely.

5.3 Implementation

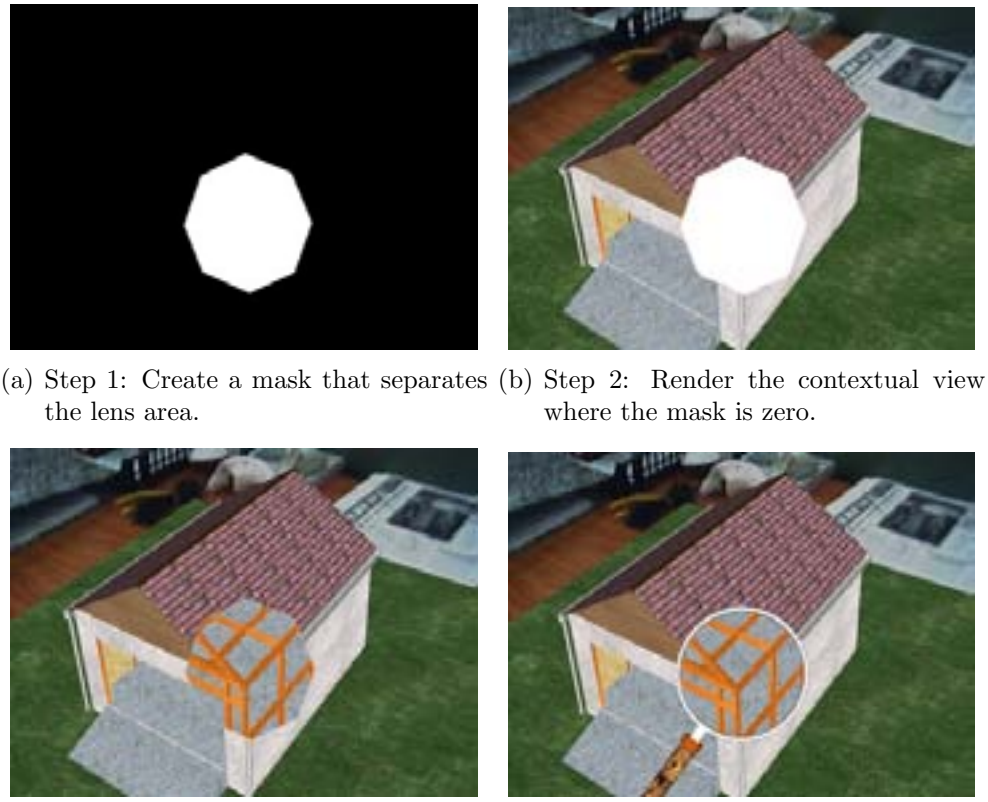
This section describes various approaches to rendering Magic Lenses in 3D virtual environments (VEs).

Stencilling is a simple approach for producing flat lenses. Stencil lenses create a mask that partitions the screen into regions of Focus and Context. This occurs in 2D screen-space, although the object that forms the mask can be manipulated with a six degree-of-freedom controller, giving the user the impression that they are using a 3D tool. We employed the stencil lens technique in our early AR Magic Lens prototypes (Looser, Billinghurst and Cockburn 2004). This approach is similar to that used in the SEAMs framework by Schaufler and Schmalstieg (1999), described in Section 2.3.2.

The stencil buffer is a memory buffer maintained by the graphics hardware, similar to the colour and depth buffers. The stencil buffer contains a value for each pixel that can be used to record a tag for that pixel location. The tag might later be used to decide whether or not rendering should be permitted for the pixel, or for some other application-specific purpose. The

stencil buffer is useful for implementing particular graphical effects such as shadows (Heidmann 1991), planar reflections (Kilgard 1999) and even image-based Constructive Solid Geometry (Kirsch and Döllner 2004).

The masking process used to implement stencil lenses is illustrated in Figure 5.3. The mask is created by rendering a 3D model of the lens object (a simple disc) into the stencil buffer resulting in a value of 1 where the lens exists and a value of 0 elsewhere (Figure 5.3(a)). The scene is then rendered normally in areas equal to 0 (Figure 5.3(b)) and in a modified state in areas equal to 1 (Figure 5.3(c)). Finally a 3D model of the magnifying glass ring and handle are drawn on top to complete the view (Figure 5.3(d)).



(a) Step 1: Create a mask that separates the lens area.

(b) Step 2: Render the contextual view where the mask is zero.

(c) Step 3: Render the focus view where the mask is one.

(d) Step 4: Render the lens tool model to complete the scene.

Figure 5.3: Producing a stencil Magic Lens effect using a mask.

The masking technique is simple, supported on all relatively modern

graphics hardware, and is a "free" operation when depth-testing is enabled (Kilgard 1999). There are, however, some limitations to this approach. Both the normal and enhanced scene need to be rendered twice. Although the stencil mask limits the number of pixels displayed for the enhanced scene, this operation occurs late in the graphics pipeline and is therefore of little use as an optimisation. It is also difficult to achieve some graphical effects (such as magnification) with this technique because the mask is generated and applied in 2D screen-space.

Dynamic Texturing (or Render to Texture, RTT) is a way to store the output of a set of rendering steps in a buffer that can later be used to texture map another object in the scene (Wynn 2002).[1] This technique is being increasingly used to produce various graphical effects, such as imposters (Forsythe 2001), reflections and refraction. Imposters are efficient 2D substitutes for complex 3D geometry. Reflection and refraction effects are most commonly used to render realistic-looking water.

More recently, dynamic texturing has become the standard way to handle custom data exchanges between the computer and the graphics card. The fast graphics processing units (GPUs) on modern video cards are being harnessed to run customised graphics and non-graphics algorithms, such as physics simulations. Many general purpose applications are possible, provided the required algorithms can be adapted to run within the constraints of the pipelined, stream processor model of the GPU (Venkatasubramanian 2003).

We investigated using dynamic textures for implementing AR Magic Lenses. Using dynamic texturing, it is possible to capture the alternate view of a 3D scene and apply it to a lens surface later in the rendering process. This approach is more flexible than the stencil buffer approach described previously. Magnification effects are easier to produce in this approach because the virtual camera that draws into the texture can be adjusted for a wider or narrower field of view. Magnification is illustrated later in Figure 5.9.

[1] Texture mapping is the method of increasing the apparent detail of a virtual surface by applying colours to it based on the pixels in a stored image. Typically the stored image is loaded from a file at runtime, but it can also be generated by a code procedure, or be some combination of the two, such as computing an image for a terrain texture map by combining a snow image with a grass image based on the terrain height at each point.

Once the scene has been rendered into the texture, the contents of the texture can be manipulated to produce Magic Lens effects at the pixel level. There are many possible image effects that could be applied at this stage. For example, all pixel colour values in the texture could be inverted to produce a negative image within the lens, or a Sobel filter could perform edge detection.

A potential drawback of this technique is that texturing only appears correct when the lens disk faces the viewer. However, this problem can be remedied by using projective texture mapping, which mimics the way a slide projector illuminates surfaces (Segal, Korobkin, van Widenfelt, Foran and Haeberli 1992). Using this technique, the lens disk can be at any angle to the viewer without the texture appearing distorted.

Clipping Planes can be used to render volumetric 3D Magic Lenses. A clipping plane divides a 3D scene into two half-spaces, one which is kept and one which is discarded. Triangles with edges that cross the plane are decomposed into smaller triangles so that a straight line edge is left. Modern graphics cards support clipping planes in hardware so they are relatively inexpensive to use. There are six clipping planes that define the OpenGL view frustum as well as at least six additional planes that are available for general use by the programmer. Six of these planes can be used to construct a cube whose volume can be rendered independently from the rest of the scene to create a Magic Lens (Viega et al. 1996). The rendering process is shown in Figure 5.4.

Rendering the content inside the cube is simple. All planes are enabled such that they discard all regions outside the cube. The scene is then rendered with the desired effect applied. This may involve hiding certain objects, or using a particular rendering style such as wireframe.

Rendering the content outside the cube is somewhat more complicated. Simply reversing the direction of the clipping planes will not invert the rendered areas. Clipping planes in OpenGL extend to infinity so that two parallel, outward facing clipping planes will clip the entire scene (see Figure 5.5). To overcome this problem, the scene must be rendered six times, once as each individual clipping plane is active on its own. This introduces a substantial amount of "overdraw", where parts of the scene are rendered more than once.

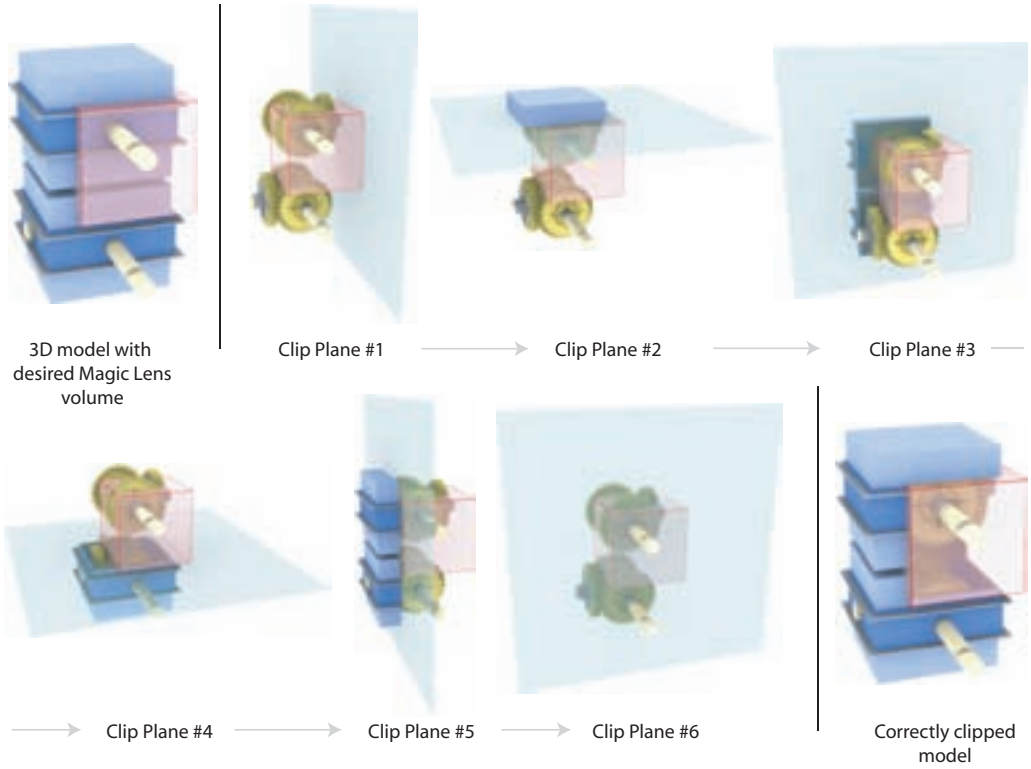
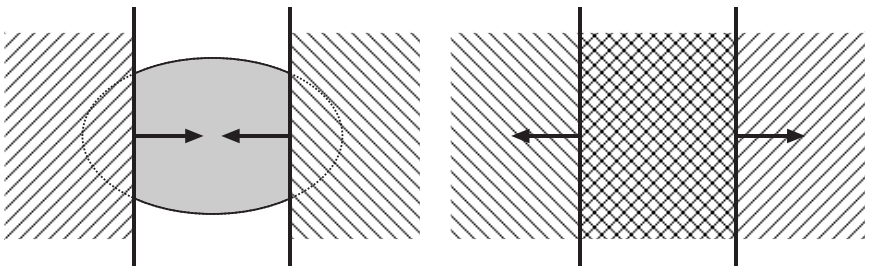


Figure 5.4: Clipping an object with clipping planes. The casing surrounding the gear assemblies is clipped to expose the gears within. Notice that in some steps of the rendering, some parts of the casing are drawn again. Overdrawing is a deficiency of this technique.

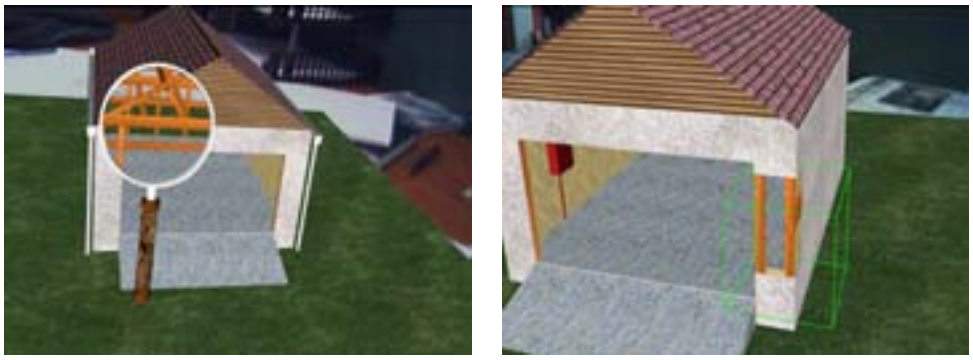


(a) Inward facing planes. Object is clipped.

(b) Outward facing planes. Entire scene is clipped.

Figure 5.5: Rendering using clipping planes. The arrows indicate the side of the plane that is kept. Diagonally shaded areas are clipped while solid areas remain. Diagram adapted from Viega et al. (1996).

Fuhrmann and Gröller (1998) describe a technique without this inefficiency, but it suffers from the problem that geometry that should be visible behind the lens is not rendered.



(a) Flat AR Magic Lens. (b) Volumetric AR Magic Lens.

Figure 5.6: The same model displayed with two AR Magic Lens techniques. Note that the lenses are being controlled by a tracked paddle, although the paddle is being obscured by the 3D graphics.

We used the clipping plane technique to implement a volumetric AR Magic Lens. Figure 5.6 shows a side-by-side view of the same 3D model visualised using firstly a flat AR Magic Lens, and secondly a volumetric AR Magic Lens.

5.4 AR Magic Lens Applications

In this section the various AR Magic Lens systems we have developed are described. All systems were designed for table-top AR environments using short range tracking. Standard desktop workstations were used for all development as consumer level equipment is more than sufficient for running the systems.

The key hardware components are the paddle, marker grid and HMD (see Figure 5.7). A paddle with an attached marker provides a tracked physical prop that is used to interact with the virtual scene on the marker grid. Like the virtual scene, the paddle's position is defined relative to the marker

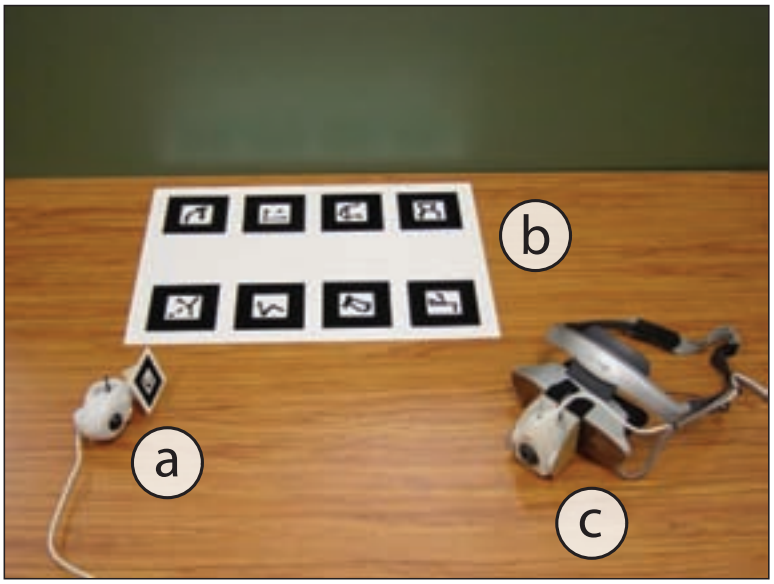


Figure 5.7: Equipment for table-top augmented reality. The paddle (a) is a hand-held mouse with a tracking marker attached. The grid of markers (b) provides the coordinate system in which the paddle operates. The HMD (c) has a camera attached for providing video see-through AR.

grid. All markers in the interface are originally tracked relative to the camera on the user's HMD, so a simple matrix transformation is required to bring the paddle marker's transformation into the marker grid's coordinate frame. This is illustrated in Figure 5.8.

A large grid of markers provides a reference plane within the real world. The user sits at a table and places the grid on the table-top before them. The virtual scene is displayed relative to this grid and therefore appears to sit on the table or hover slightly above it. The user can look at the scene from different directions by physically moving their head and body, or they can manipulate the grid of markers itself, such as rotating it or moving it nearer for a closer look.

Finally, a Head Mounted Display (HMD) with an attached camera provides the user with the video see-through AR experience, described in detail in Section 2.4.2.

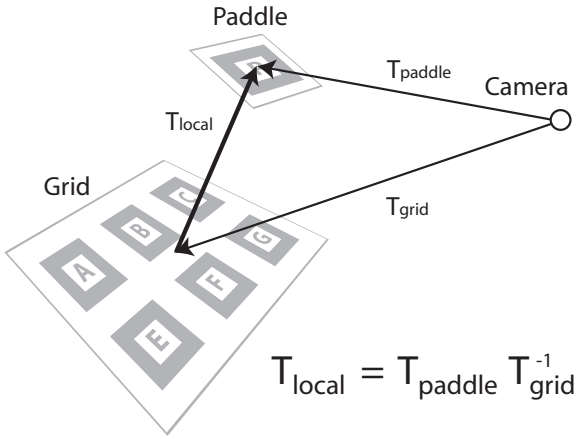


Figure 5.8: The transformations involved in calculating the paddle position relative to the grid of markers.

The key software component in the majority of the applications is osgART, a software library created to aid the development of AR applications. The development of this library was considered necessary because the effort required in developing with raw OpenGL made rapid prototyping of ideas too time consuming. Much in the same way as a good user interface toolkit for desktop applications can aid interface designers, the same is true when working in augmented reality.

osgART stands for Open Scene Graph Augmented Reality ToolKit. Open Scene Graph is a scene-graph library built on top of OpenGL that simplifies the development of graphical applications and enhances performance (Osfield 2007). osgART links video capture devices, computer-vision based tracking libraries, and Open Scene Graph to provide the necessary functionality for video see-through AR. The development and features of osgART are described in detail in Chapter 6.

The Magnifier application aims to accurately reproduce the behaviour of a real magnifying glass in AR. This application uses dynamic texturing to render the view within the lens. A virtual camera captures an image of the 3D scene from the current location of the lens, determined by the position

of the handheld paddle over the marker grid. The captured image is then textured onto the lens object (a simple disk) to create the illusion that the user is seeing "through" the lens.

To enhance the effect further, the field of view (FOV) of the virtual camera is also manipulated, using the distance the user holds the lens away from their eyes as an input parameter. Therefore, as the user moves the lens away, objects seen through the lens grow smaller, and as they bring the lens closer, objects are magnified.

As the magnification effect is entirely virtual, it is possible to experiment with other mappings between lens distance and magnification factor. For example, the relationship could be reversed so that the scene actually grows smaller as the lens is brought near. Although this is counter-intuitive for magnification, it means that in a single motion, the user can expand the area of focus (the lens area gets larger due to perspective) and also see more of the scene through the lens (due to reduced magnification).

The overall effects achieved with the interface are compelling and realistic. One unresolved challenge is the implementation of a lens that seamlessly magnifies both the virtual content of the scene and the live video image from the video camera.

(a) A garden scene.

(b) A terrain scene.

Figure 5.9: Examples of the Magnifier lens.

The Globe Browser provides the user with a simple tool for comparing global datasets. In this interface, a virtual globe appears before the user, suspended in space above the tracking grid. The grid can be rotated to view the globe from different directions, or the user can move around the grid. It is also possible to freely rotate the globe itself using the trackball integrated into the handheld controller.

A library of global maps is loaded into memory. The maps each present a different dataset or view of the Earth. Some examples include SeaWiFS Chlorophyll data, the NASA Blue Marble Image, and the Earth at Night. These three examples are shown in Figure 5.10.

Figure 5.10: The Globe Browser

The globe browser uses a Magic Lens based on the stencil (masking) technique. One map is shown within the Magic Lens and another is shown outside. Two datasets can be compared over a region of the planet by simply placing the Magic Lens over it. The user clicks one button on the handheld controller to move to the next dataset, or another button to apply the current dataset globally. In this way, it is possible to view any two of the available datasets at the same time.

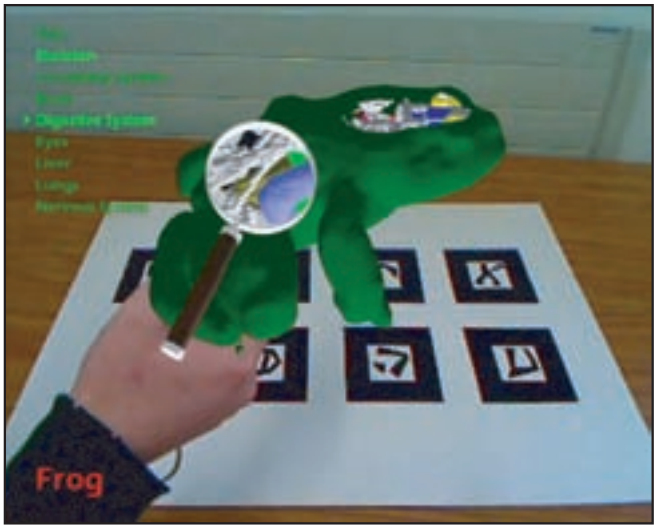
The X-Ray Vision Interface allows the user to explore the internal structures of 3D models. Many 3D models are comprised of nested components making it difficult to understand their structure. Examples include the human body, buildings, and engineered machines. There are numerous ways to visualise these complex models, such as exploded views where the parts are moved apart from each other, or simpler approaches like semi-transparency and wireframe views. An AR Magic Lens interface can also be used to visualise these types of models.

In the X-Ray Vision Magic Lens interface, complex models are loaded as a series of components. Each component can be interactively enabled or disabled within the Magic Lens view, while the full 3D scene remains intact outside the lens region as context. This type of interaction is particularly compelling and is reminiscent of educational children's books such as the Incredible Cross Sections series by Platt and Biesty (1992).

Figure 5.11 illustrates three examples where the X-Ray Magic Lens has been used. Two of these examples deal with anatomy, where complex 3D objects are densely packed together. The ability to identify, show, and hide individual internal systems (e.g. lymphatic, circulatory) has great potential as an educational aid, and the fact that the area outside the lens continues to display the familiar anatomical form supports understanding. The third example demonstrates the same technique applied to a building, with the addition of optical magnification.

Information enhancement was also investigated with the AR Magic Lens. In a similar way to the Globe Browser described previously, the Magic Lens can provide access to additional layers of information. As described in the scenario of the 2D Magic Lens experiment in Chapter 4, Geographical Information Systems (GIS) deal with massive amounts of layered information. AR has already been explored as a potential platform for interacting with this data. For example, Hedley, Billinghurst, Postner, May and Kato (2002) investigate ways of bringing GIS datasets into the real world and displaying them over physical maps. They even go so far as to suggest that a Magic Lens technique could be suitable for this type of interface. Figure 5.12(a) provides a simple example of examining two GIS datasets simultaneously using an

(a) Investigating human anatomy.

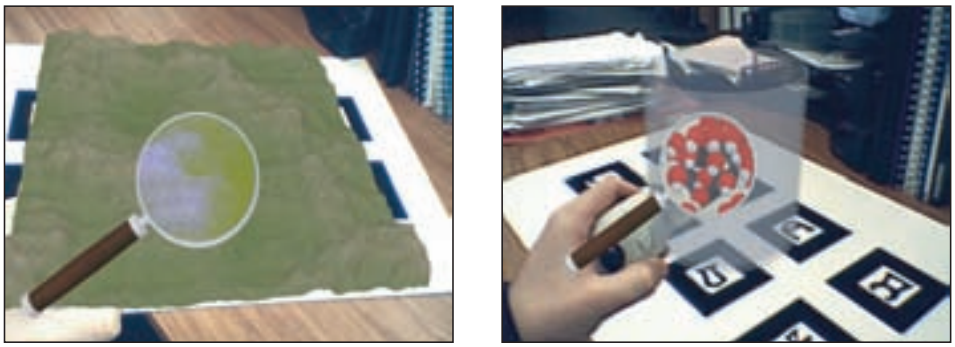


(b) Investigating frog anatomy.

(c) Viewing the internal structure of a house (with optical magnification).

Figure 5.11: Examples of the X-Ray Vision Interface.

AR Magic Lens. Another potential application for AR Magic Lenses is as an educational aid. Figure 5.12(b) illustrates an application where the lens "magnifies" the unidentifiable contents of a beaker to reveal the molecular structure of the liquid.



(a) Viewing abstract GIS data in the context of virtual terrain.

(b) A Magic Lens view of molecules in a chemistry scenario.

Figure 5.12: Applications of the Magic Lens for Information Magnification.

Rendering Effects within AR Magic Lenses manipulate the way in which 3D objects are displayed. Modern rendering APIs, such as OpenGL and DirectX, give the graphics programmer great control over the appearance of surfaces, particularly through the use of vertex and pixel shaders on the GPU. A Magic Lens can be used as a creative way to compare and investigate different rendering styles.

A simple version, the Wireframe Magic Lens shown in Figure 5.13(a), simply alters the OpenGL rendering state to force line drawing rather than filled polygons. The NPR (Non-Photorealistic Rendering) Magic Lens, shown in Figure 5.13(b), demonstrates a more complicated approach involving cartoon rendering.

TankWar is an AR strategy game in the style of traditional table-top board games. The development and evaluation of TankWar was the subject of Trond Nilsen's masters thesis (Nilsen 2006). I developed the 3D graphics component of the original version of TankWar.

(a) Simple wireframe Effect.

(b) Non-photorealistic rendering using cartoon shading.

Figure 5.13: Applications of the Magic Lens for Rendering Effects.

The TankWar game involved two players directing teams of tanks on a virtual battlefield that appeared above a real table. The goal was either to capture all objective points on the map, destroy all of the opponents tanks, or work together against a computer controlled player. Tanks automatically fired on enemy tanks within range, so the objective was to manoeuvre your own team into the best positions.

An AR Magic Lens tool was the primary interaction tool for players of TankWar. Figure 5.14(a) shows the lens being used to magnify the battle-field. A crosshair in the centre of the lens view was used to select tanks, designate targets on which to fire, and to choose locations for the tank to move to. Using the AR Magic Lens as a selection tool is explored and evaluated in detail in Chapter 7.

In the co-operative mode of the game the view frustum of the other player's lens was displayed as a semi-transparent volume projecting into the 3D world (see Figure 5.14(b)). This was a valuable cue for the players to know where the other player was currently focussing their attention.

A further feature of TankWar was the ability to transition into the battle-field and experience it from a first-person perspective. This type of interaction was inspired by the same feature within the MagicBook of Billinghurst et al. (2001). When a player entered the battle-field, the video of the real world faded away and was replaced by a completely virtual environment.

However, the lens frustum of the other player was still visible, as shown in Figure 5.14(c), providing the user with some situational awareness.

One final feature of the Magic Lens in TankWar was the ability to use it like a camera to save particular views. In TankWar, the "photos" were uploaded to a website that also showed current statistics of the game in progress.

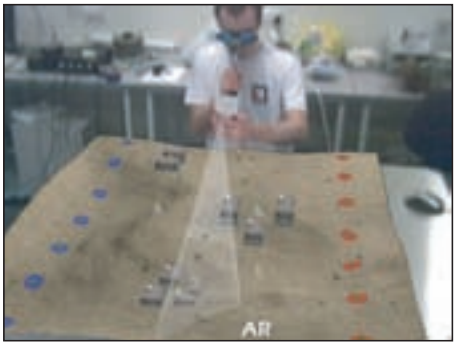
TankWar was demonstrated at several public venues and conferences (see Figure 5.14(d)). Nilsen (2005) went on to present the game at GenCon Indy where about 300 people played the game. Questionnaires were collected from 230 of the players. Although players found it difficult to see detail through the HMDs, found the equipment heavy and difficult to adjust, and gave a low overall rating for ease-of-use, the game was still greeted with acclaim.

Fisheye Effects have also been explored in augmented reality. Although not strictly a Magic Lens, the fisheye distortion techniques described in Section 2.2.1 are also an interesting area of research for TAR interfaces. We applied a graphical fisheye distortion to a map, used a physical cube as a tangible controller, and video see-through AR for visualisation.

This interface was inspired by the Tangible Augmented Street Map of Moore and Regenbrecht (2005), which also aims to present a large map using a tangible cube, but does so by displaying individual square map tiles on each side of the cube. As the cube is rotated, the direction of movement is detected and new tiles appear. Therefore the rolling motion is analogous to panning across a map, albeit in discrete steps.

In the fisheye version of this interface, discrete tiles are replaced with a seamless map that pans smoothly as the cube is rolled in the user's hands. The details of the implementation of this system are provided in Figure 5.15. An additional feature is a cursor that indicates which point on the map lies directly in the centre (and therefore the focus) of the user's view. This information could be used to trigger events such as additional pop-up information about the location being viewed.

(a) User s view with AR Magic Lens tool.



(b) Collaborator s view. Note the view volume extending from the other user s AR Magic Lens tool.

(c) Virtual Reality view with other player s lens frustum visible.

(d) External view.

Figure 5.14: The TankWar Augmented Reality game. Images courtesy of Trond Nilsen.

5.5 Summary

In this chapter we have presented the motivation for AR Magic Lenses. Implementation techniques for flat and volumetric 3D Magic Lenses were explained, and applied in a large range of Augmented Reality interfaces. Each type of interface demonstrated a way in which AR Magic Lenses can be used, including applications in architecture, GIS, entertainment and education.

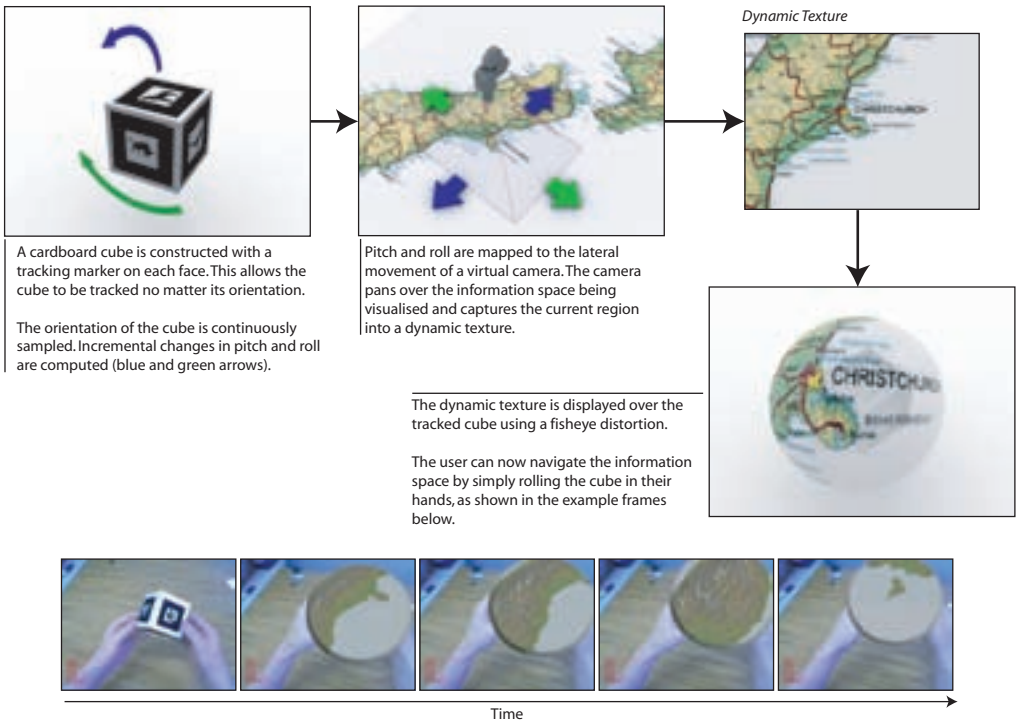


Figure 5.15: Tangible Cube inspired by Moore and Regenbrecht (2005).

Chapter 6

osgART

This chapter describes osgART, a software library developed to aid the development of AR applications. The development of this library was considered necessary because the effort of working in raw OpenGL made rapid prototyping of ideas too time consuming. Much in the same way as a good user interface toolkit for desktop applications can aid interface designers, the same is true when working in augmented reality.

Figure 6.1: The osgART library logo.

This chapter begins with a necessary description of scene graphs, and the introduction of one particular scene graph library, Open Scene Graph. Next, the process of modifying Open Scene Graph to incorporate video see-through Augmented Reality capabilities is described. The resulting software framework, osgART, is then demonstrated with some simple example applications and source code.

6.1 Scene Graphs

A scene graph is a tree-like structure used to organise a virtual world. The tree is composed of nodes which form a hierarchy with a single node at the root. Nodes can define transformations, geometry, groupings and material properties, among other things. A rendering is produced by traversing the tree in a depth-first manner, starting at the root. Scene graphs can greatly simplify the development of virtual worlds by handling many of the difficult aspects of 3D rendering. An example scene graph is shown in Figure 6.2.

Rendering performance is increased primarily through culling and sorting. Culling refers to discarding unseen objects early in the rendering process to reduce processing load. Objects may become candidates for culling when they move off-screen, are occluded by other objects or become smaller than a few pixels on the screen. Sorting refers to managing the order in which objects are processed to minimise expensive rendering state changes and to ensure graphical quality. For example, the sorting manager must consider that drawing all objects of the same material in a row is better than switching for each object, and that transparent objects must typically be drawn from farthest to nearest to produce the correct blending of colours.

Open Scene Graph(Osfield 2007) is an open source scene-graph implementation built on top of OpenGL. It is written in object-oriented C++ with attention to design pattern principles. For example, visitor objects play a crucial role in updating and interrogating the scene graph. Open Scene Graph has an active development community and is used in numerous graphics domains including simulation, games, online-chat, military and industrial projects.

6.2 Open Scene Graph and ARToolKit Integration

The ARToolKit was integrated with Open Scene Graph to add video see-through augmented reality capabilities. The result, named osgART, is a robust and extensible platform for building augmented reality applications (Looser et al. 2006, Grasset, Looser and Billinghurst 2005). Although osgART was originally created by myself, its current state is the result of

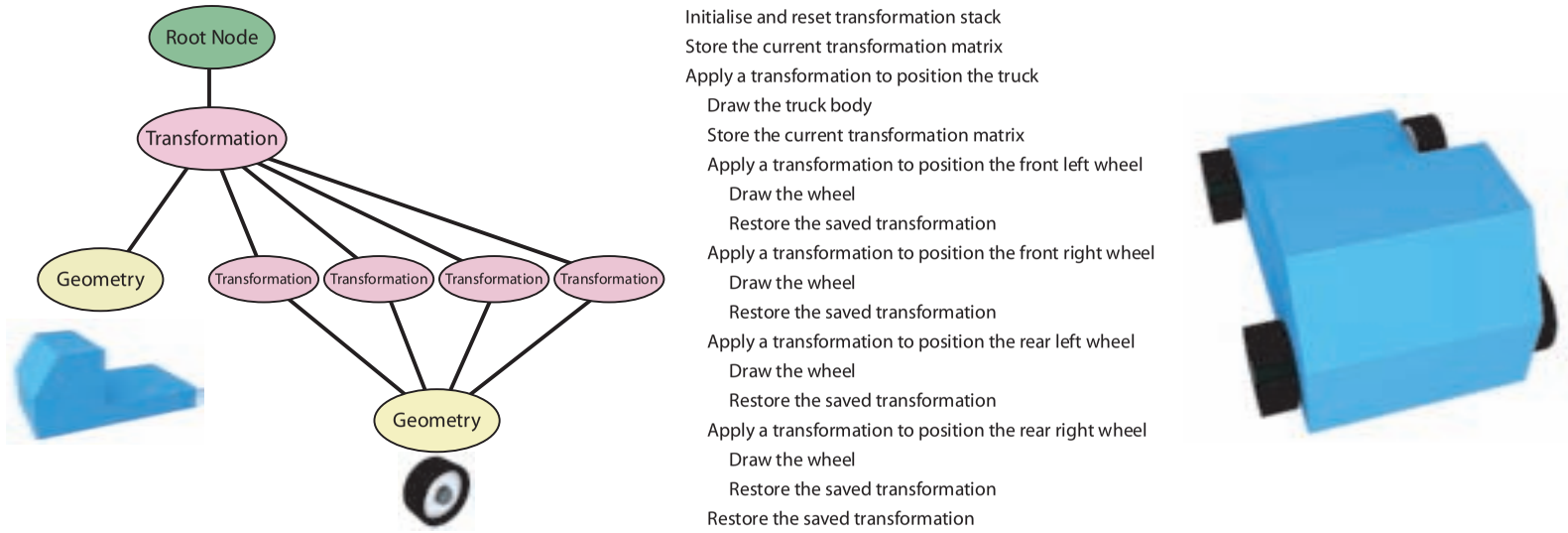


Figure 6.2: An example scenegraph for a simple truck. The diagram on the left shows the scene graph structure. The code in the middle is a procedural version created when the scene graph is traversed. The image on the right shows the resulting rendering of the scene graph.

the additional development efforts of Raphaël Grasset and Hartmut Seichter (post-doctoral researchers at HIT Lab NZ) and Philip Lamb (CTO, ARToolWorks).

The original implementation of osgART was tightly coupled with the ARToolKit tracking library and video capture system. It has since been generalised through a plugin architecture which allows it to support new tracking libraries and video capture devices. For example, it can operate with the BazAR[1] object tracking library and the PointGrey[2] range of high-quality video capture products.

6.2.1 Implementation of osgART

This section describes the implementation of osgART. This description is supported by Figure 6.3 which shows the structure of osgART. The main components of the system are a video source, a tracker, and a scene graph to render the output.

The video source captures frames from an attached camera (or from a pre-recorded video file) and makes them available to the tracking component and scene graph. The tracker requires frames in order to locate markers and calculate transformations. The scene graph requires frames to display the real world behind the virtual objects.

When the tracker locates a marker and computes its transformation, that information is transferred into nodes within the scene graph. The Open Scene Graph MatrixTransform class was subclassed to create the ARTTransform class which automatically updates itself based on the transformation of an associated tracking marker. Therefore, geometry placed beneath the ARTTransform in the scene graph will appear anchored to its marker. If the marker is unavailable for some reason (such as the marker is occluded) then the ARTTransform sets its NodeMask to zero, which hides the node during the rendering traversal, making it and its children invisible.

The 3D graphics are displayed within a perspective projection, deter-

[1] BazAR vision-based fast detection library, `http://cvlab.epfl.ch/software/bazar/`, online as of September 2007.

[2] Point Grey Research, `http://www.ptgrey.com/`, online as of September 2007.

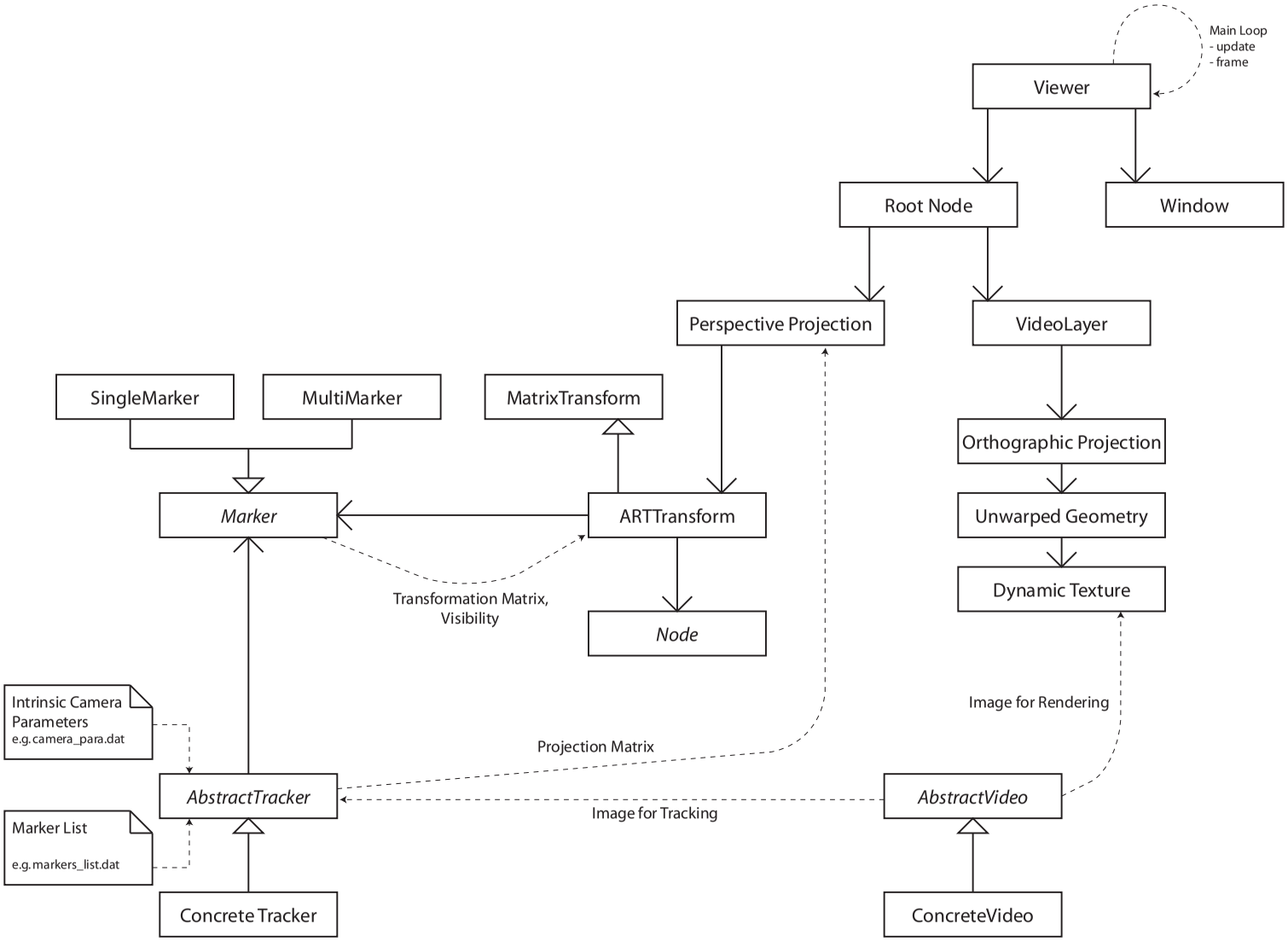


Figure 6.3: The structure of the osgART scenegraph.

mined by a projection matrix provided by the tracker. The tracker creates this matrix using intrinsic camera parameters determined during an offline calibration step for the specific camera being used. These parameters are necessary for the tracker to accurately track markers, and the correct projection matrix is required for the resulting transformations to align with the live video.

The video frames from the camera are continuously uploaded into a texture within the scene graph. A fullscreen quad is then mapped with this dynamic texture and placed within an orthographic projection. This branch of the scene graph is set to render first so that the video always appears behind the 3D graphics. The simple quad can optionally be replaced with a grid mesh, where the vertices are automatically adjusted to compensate for distortion in the camera's lens (again based on information collected during calibration). This functionality is encapsulated within the VideoLayer class in osgART.

The programmer can instantiate as many ARTTransform objects as they desire. Each must be associated with a tracking marker. The collection of markers available is configured within a simple text file. Markers can either be standalone, or grouped together. The advantage of tracking a group of markers is that it provides redundancy for when one or more of the markers is occluded or out of view.

6.2.2 osgART Example Code

The first code block demonstrates a minimal osgART application. The system is initialised, and a blue cube is programmed to appear on a single marker, as shown in Figure 6.4.

```
#include <osgART/Foundation>
#include <osgART/ARTTransform>
#include <osgART/VideoLayer>
#include <osgART/ARSceneNode>
#include <osgART/PluginManager>
#include <osgART/Viewer>
#include <osg/ShapeDrawable>

int main(int argc, char* argv[]) {
                                                                    10
    // Preload the tracker plugin
    if (!osgART::PluginManager::instance()->load("osgart_tracker_artoolkit"))
        exit(-1);
```

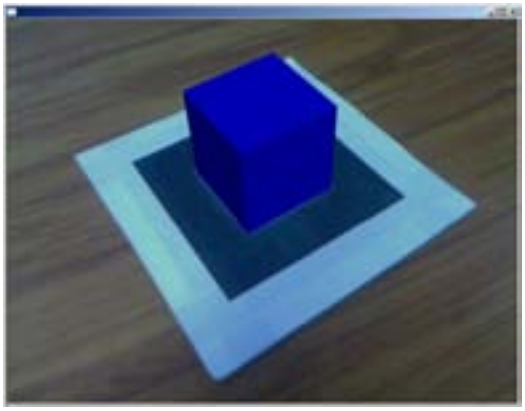


Figure 6.4: A simple AR application created with osgART.

```
// Preload the video plugin
if (!osgART::PluginManager::instance()->load("osgart_video_artoolkit"))
    exit(-1);

// Instantiate a viewer and create a root node
osgART::Viewer viewer;
osg::ref_ptr<osgART::ARSceneNode> root = new osgART::ARSceneNode;
viewer.setSceneData(root.get());

// Create an instance of the video plugin
osg::ref_ptr<osgART::GenericVideo> video =
    dynamic_cast<osgART::GenericVideo*>(osgART::PluginManager::instance()->get("video_artoolkit"));

// Ensure the video is valid
if (!video.valid()) {
    osg::notify(osg::FATAL) << "Could not initialize video plugin!" << std::endl;
    exit(-1);
}

// Configure and open the video
video->setFlip(false,true);
video->open();

// Create an instance of the tracker plugin
osg::ref_ptr<osgART::GenericTracker> tracker =
    dynamic_cast<osgART::GenericTracker*>(osgART::PluginManager::instance()->get("tracker_artoolkit"));

// Ensure the tracker is valid
if (!tracker.valid()) {
    osg::notify(osg::FATAL) << "Could not initialize tracker plugin!" << std::endl;
    exit(-1);
}

// Connect the video and the tracker
if (!root->connect(tracker.get(),video.get())) {
    osg::notify(osg::FATAL) << "Error connecting video with tracker!" << std::endl;
    exit(-1);
}

// Create a group to hold the scene
osg::Group* sceneGroup = new osg::Group();
sceneGroup->getOrCreateStateSet()->setRenderBinDetails(2, "RenderBin");

// Create a video background for the live video
```

```
    osg::ref_ptr<osgART::VideoLayer> videoBackground = new osgART::VideoLayer(video.get() , 1);
    videoBackground->init();
    sceneGroup->addChild(videoBackground.get());

    // Create a projection from the tracker's projection matrix
    osg::Projection* projectionMatrix = new osg::Projection(osg::Matrix(tracker->getProjectionMatrix()));
    projectionMatrix->addChild(sceneGroup);
    root->addChild(projectionMatrix);

    // Create a marker and ensure it is valid
    osg::ref_ptr<osgART::Marker> marker = tracker->getMarker(0);
    if (!marker.valid()) {
        osg::notify(osg::FATAL) << "No Marker defined!" << std::endl;
        exit(-1);
    }
    marker->setActive(true);

    // Create a transformation node for the marker
    osg::ref_ptr<osg::MatrixTransform> markerTrans = new osgART::ARTTransform(marker.get());
    markerTrans->getOrCreateStateSet()->setRenderBinDetails(5, "RenderBin");
    sceneGroup->addChild(markerTrans.get());

    // Add a blue box below the transformation
    float boxSize = 40.0f;
    osg::ShapeDrawable* sd = new osg::ShapeDrawable(new osg::Box(osg::Vec3(0, 0, boxSize / 2.0f), boxSize));
    sd->setColor(osg::Vec4(0, 0, 1, 1));
    osg::Geode* geode = new osg::Geode();
    geode->addDrawable(sd);
    markerTrans->addChild(geode);

    // Start the video and viewer
    video->start();
    viewer.realize();

    // Mainloop
    while (!viewer.done()) {
        viewer.frame();
    }

    // Cleanup
    video->stop();
    video->close();

}
```

The second code block, in Figure 6.6, shows that another marker can be easily added by simply instantiating another ARTTransform, and assigning a different marker to it. The result is shown in Figure 6.5.

6.3 Summary

This chapter described osgART, a software library developed to support the development of AR Magic Lenses. osgART integrates video see-through Augmented Reality libraries, such as ARToolKit, with Open Scene Graph. Using a scenegraph with ARToolKit has been suggested before (such as by Haller, Hartmann, Luckeneder and Zauner (2002)) and a similar integration project, OSGAR, was recently undertaken by Coelho, MacIntyre and Julier (2004).

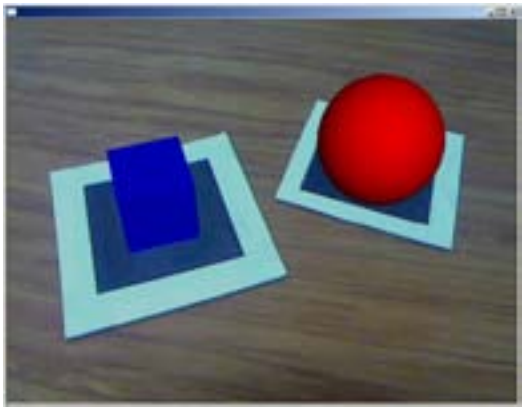


Figure 6.5: Multiple independently tracked markers with osgART.

```
// First marker
osg::ref_ptr<osgART::Marker> markerA = tracker->getMarker(0);
markerA->setActive(true);
osg::ref_ptr<osg::MatrixTransform> markerTransA = new osgART::ARTTransform(markerA.get());
markerTransA->getOrCreateStateSet()->setRenderBinDetails(5, "RenderBin");
sceneGroup->addChild(markerTransA.get());

// Add a blue box below the transformation
float boxSize = 40.0f;
osg::ShapeDrawable* sdA = new osg::ShapeDrawable(new osg::Box(osg::Vec3(0, 0, boxSize / 2.0f), boxSize));
sdA->setColor(osg::Vec4(0, 0, 1, 1));
osg::Geode* geodeA = new osg::Geode();
geodeA->addDrawable(sdA);
markerTransA->addChild(geodeA);


// Second marker
osg::ref_ptr<osgART::Marker> markerB = tracker->getMarker(1);
markerB->setActive(true);
osg::ref_ptr<osg::MatrixTransform> markerTransB = new osgART::ARTTransform(markerB.get());
markerTransB->getOrCreateStateSet()->setRenderBinDetails(5, "RenderBin");
sceneGroup->addChild(markerTransB.get());

// Add a red sphere below the transformation
float sphereSize = 40.0f;
osg::ShapeDrawable* sdB = new osg::ShapeDrawable(new osg::Sphere(osg::Vec3(0, 0, sphereSize / 2.0f), sphereSize));
sdB->setColor(osg::Vec4(1, 0, 0, 1));
osg::Geode* geodeB = new osg::Geode();
geodeB->addDrawable(sdB);
markerTransB->addChild(geodeB);
```

Figure 6.6: Two independent markers and transformations.

The focus of that project was to investigate registration error in augmented reality interfaces, and how the detrimental effects of that error can be reduced by adapting the display at runtime. In contrast, the main goal of osgART is to simplify the development process for AR interface builders.

osgART has been used in a variety of projects and has been released under both open-source and commercial licenses.

Chapter 7

AR Magic Lens Selection Evaluation

7.1 Introduction

Like all user interfaces, those based on Tangible Augmented Reality (TAR) require a set of basic interaction techniques the user can engage to perform actions in the interface. In 3D environments, such as AR applications, interaction techniques are generally categorised into selection, manipulation, navigation and system control (Bowman, Kruijff, Joseph J. LaViola and Poupyrev 2004).

Selection is the process of identifying an object, or set of objects, so that they can be interacted with. Once a selection is made, the selected objects can respond to manipulation, which includes moving the object, orienting or scaling it, as well as any other application-defined actions. Navigation involves the adjustment of the view position and orientation within the virtual environment and is often broken down into Travel (the means of moving from viewpoint A to viewpoint B) and Wayfinding (the cognitive issues related to successfully completing the movement). System control covers actions that change the interaction mode or system state, such as switching between a selection tool and a manipulation tool.

In this chapter, selection within a table-top Augmented Reality environment is evaluated. Selection is fundamental because it precedes many other actions. Selection can determine the objects users wish to manipulate and the locations to which they wish to navigate. For example, to orient and position a house in an immersive urban planning tool, the user must first select the component (or set of components) that make up the house from the set of all components in the scene. Only then will subsequent manipulations be applied to the correct objects.

Performance with a particular selection technique can be measured via a set of metrics such as time taken to make the selection, accuracy of selection, and number of errors. Qualitative measures such as mental and physical effort are also important. The features of the environment that can affect selection performance include the target object's size and distance (objects that are small, due either to geometrical size or range from the viewpoint, are more difficult to select), the density of surrounding objects (distractors) and visibility (occluders) (Bowman et al. 2004).

Previous chapters in this thesis have reported on the use of virtual lenses as a visualisation technique in AR. Following the TAR approach, a virtual magnifying glass is attached to a physical handheld prop (Looser et al. 2004). The user's view is now partitioned into a primary view and a secondary view seen through the lens of the magnifier. The lens view is generalised in the style of Magic Lenses (Bier et al. 1993), such that the content seen through it, and the style in which it is visualised, can be configured independently of the primary view.

The combination of Tangible Augmented Reality and Magic Lenses has many intriguing applications. Previous research has begun to explore these applications, however until now there have been few formal evaluations conducted. For example, Tangible AR Magic Lenses were used as the primary interaction method for the table-top augmented reality game TankWar, described in Section 5.4. The lens tool was used to select and direct tanks around a virtual battlefield however no evaluation of selection or pointing performance was conducted.

It is important to benchmark new interaction techniques so that they can be described and understood within the context of existing techniques. In this chapter the filtering and visualisation aspects of the tool are temporarily set aside and we concentrate on the extent to which the virtual lens supports object selection. Selection is an important addition to the virtual lens tool because it expands its role in the user interface beyond visualisation support, to that of a more general purpose instrument.

The scope of this experiment is local tabletop AR. We are investigating selection techniques for interaction with content within arm's reach. It is possible, however, that some of these techniques will work in other environ-

ments, such as AR on mobile devices or for wide-area outdoor AR.

7.2 Related Work

7.2.1 Object Selection

A large number of selection techniques have been proposed and implemented for immersive virtual environments. Although the field is large, it is generally well understood due to the many rigorous evaluations that have been carried out (e.g. Bowman, Johnson and Hodges (2001)) and the development of taxonomies to structure our understanding. For example, the classification-by-metaphor taxonomy of Poupyrev, Ichikawa, Weghorst and Billinghurst (1998) is treated as a de facto standard. At the highest level, it partitions the space of selection techniques into either *exocentric*, those that operate from a third-person perspective, or *egocentric*, those that operate from a first-person perspective.

Whether a technique is considered egocentric or exocentric depends on the context in which it is used. A good example of the exocentric case is World-in-Miniature (WIM) (Stoakley et al. 1995). The WIM technique displays a small copy of the virtual environment in the hand of the user, who can then use it as a proxy for object selection and manipulation and as a navigation aid. In this interface, the low-level selection operation is achieved through pointing, but the interaction technique is considered exocentric because of the user's third person view of their target.

Egocentric techniques are generally more interesting for AR because AR interfaces are anchored to the user's real view, and therefore favour a first-person perspective. Egocentric techniques are further categorised into those that follow the Virtual Hand metaphor and those that follow the Virtual Pointer metaphor. Virtual Hand techniques involve directly touching target objects (either through close proximity or collision) whereas Virtual Pointer techniques involve indirectly designating targets from a distance (such as by a virtual ray).

The most basic Virtual Hand technique is a direct mapping between real and virtual hand motion. This mapping can be manipulated to create new techniques, such as Go-Go (Poupyrev, Billinghurst, Weghorst and Ichikawa

1996), which introduces a non-linear relationship between the offset of the user's physical hand and the offset of the virtual hand to greatly increase the user's reach within the virtual environment. In AR, Virtual Hand techniques can be implemented by tracking the user's fingers (Piekarski 2004), or through a tracked handheld tool.

Ray-casting is the simplest Virtual Pointer technique. A ray originating at the user's virtual hand shoots out in the direction they are pointing. Typically the first object to be hit by the ray is selected, however often selecting the first object is not ideal. Recently Grossman and Balakrishnan (2006) explored various disambiguation mechanisms for multiple target intersections for 3D volumetric displays, finding an enhancement called *Depth Ray* to perform faster and with fewer errors.

A weakness of ray-casting is that a slight change in angle at the origin of the ray equates to an increasingly large change in angle along the ray. Therefore, selecting small or distant objects can be difficult. There are several variations of ray-casting that address this problem. Cone-casting (Liang and Green 1994) uses a cone to select objects based on their relative distances from the ray. Objects that are far from the user are allowed to be further from the ray and still be selected. Shadow Cone (Steed and Parker 2004) is a further refinement that selects objects that remain continuously within the cone while selection is active. This provides the user with finer control for complex selection tasks with a high level of occlusion as they can modify their selection on the fly.

Aperture selection, illustrated in Figure 7.1, is a cone-based technique where the cone originates at the user's eye-point and passes through a circle defined by a tool held in the user's outstretched hand (Forsberg et al. 1996). The direction of the cone is controlled by moving the tool left, right, up and down, and the spread of the cone is controlled by moving the tool nearer or farther away.

Image Plane selection techniques, introduced by Pierce, Forsberg, Conway, Hong, Zeleznik and Mine (1997), reduce 3D object selection to a 2D task by operating on the 2D projection of the 3D scene. The Sticky Finger version of this technique, shown in Figure 7.2, casts a ray from the user's eye, through their finger on the 2D projection plane, and out into the scene,

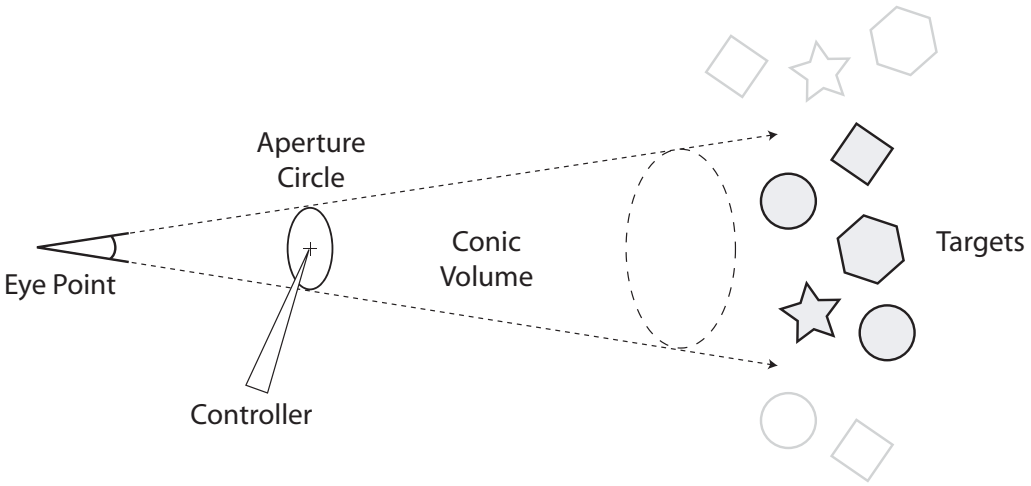


Figure 7.1: The Aperture selection technique. A cone extending from the user's eye through the circle of the handheld tool is used to make selections. Image recreated based on that of Forsberg et al. (1996).

selecting the first object to be hit. This can be considered a limiting case of aperture selection, where the spread of the cone is effectively zero.

Image plane techniques are common in desktop 3D environments, where the mouse is the primary device for interaction. The mouse cursor operates in 2D screen-space and when the mouse button is clicked, a ray can be cast through the cursor into the 3D world. This approach is standard in 3D modelling tools and real-time strategy computer games, for example. Another common approach is to fix a selection cross-hair in the centre of the screen and rely on the user's movements to bring target objects into view. As the user navigates through the virtual environment, any object falling in line with the cross-hair becomes a candidate for selection. This is standard in first-person computer games, virtual walk-throughs and so on.

Users may want to select single or multiple objects. Sometimes multiple objects can be easily selected because they are located together and can be divided from the rest by a drawn box or lasso for example. At other times a selection must be built up from many consecutive single selections, or by making a large group selection and then removing unwanted objects. These challenges are frequently encountered in standard 2D user interfaces, where

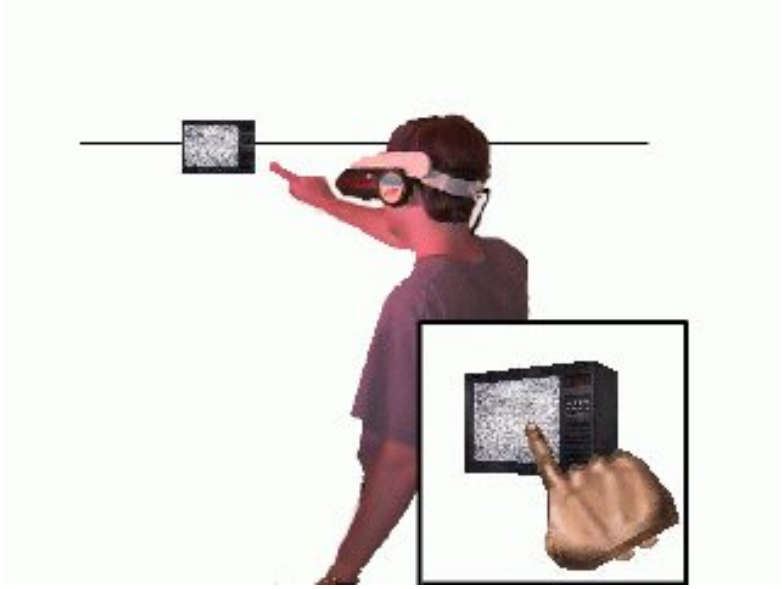


Figure 7.2: The Sticky Finger image plane selection technique of Pierce et al. (1997). With this technique the user simply points to the object of interest. The inset shows the user's view. Image courtesy of Andrew Forsberg.

long lists of items or cluttered windows of icons are common.

For a more detailed survey of selection techniques for 3D virtual environments refer to Bowman et al. (2004). Most techniques designed for VR are easily adapted for AR interfaces. In contrast, some selection techniques have emerged as a result of features or needs specific to AR. For example, in AR interfaces that use fiducial marker tracking, such as ARToolKit (Kato and Billinghurst 1999), the loss of tracking that occurs when the user covers a marker with their finger can be used to indicate selection (Lee, Billinghurst and Kim 2004). This type of interaction is shown in Figure 7.3.

Boeck, Weyer, Raymaekers and Coninx (2006) carried out a formal evaluation comparing three selection techniques (direct touch, ray-casting and aperture) in a virtual environment. They tested each technique's performance when controlled by both the dominant or non-dominant hand and found aperture selection to be the fastest technique, even performing faster in the non-dominant hand than the other two techniques in the dominant hand.

Figure 7.3: Occlusion selection technique for marker-based AR by Lee et al. (2004). In this example, a rudimentary slider is implemented by determining which markers in the row are currently blocked from the camera. Image courtesy of Gun Lee.

Fitts Law

Fitts' Law is a universally accepted law that relates human movement time to the distance to and size of a target (Fitts 1954). The law has been applied extensively in human-computer interaction research, where it is used as a tool to evaluate the usability of user interfaces and guide the development of hardware input devices (MacKenzie 1995). The general form of the Fitts' Law equation is:

$$MT = a + b \times IoD \tag{7.1}$$

where MT is the movement time in seconds, a and b are experimentally derived constants, and IoD is the index of difficulty in bits. The index of difficulty is a measure of how difficult a target is to select, formulated with respect to the target's size and distance. The index of difficulty for selecting 2D targets is calculated as:

$$IoD = \log_2 \left(\frac{A}{W} + 1 \right) \tag{7.2}$$

where A is the amplitude, or distance to the target, and W is the width of the target. By experimenting with varying values for the IoD, it is possible to derive values for the constants a and b from Equation 7.1. Fitts' Law describes a linear relationship, where a gives a base value, and b gives the slope. In the task of selecting a target, a represents the cognitive and motor

preparation time and b is a measure of hand-eye coordination.

The Index of Performance (IoP), also known as *bandwidth*, in Fitts' Law is a measure of how much information can flow through a particular channel. This is analogous to bandwidth in the context of information theory. In this case, the channel is the user's limb and the pointing device. The IoP, measured in bits per second, is defined as the reciprocal of b, the slope of the regression line.

$$IoP = \frac{1}{b} \tag{7.3}$$

7.3 Experimental Design

The goal of this experiment is to compare a selection technique built for existing AR virtual lens interfaces (described in Section 5.4) with two traditional techniques based on the approaches of virtual hand and virtual pointer: direct touch and ray-casting. This evaluation is important because selection is a particularly common activity in virtual environments and therefore is a prime target for optimisation.

7.3.1 Apparatus

The experiment was run on what has become a familiar desktop AR configuration: a webcam attached to a head-mounted display, connected to a computer running ARToolKit based software. In this case, the camera was a Logitech Notebook Pro (640x480 pixels at 30FPS), the HMD was an eMagin Z800 (800x600 pixels, 40° field of view) and the computer was a 3.2GHz PentiumD Shuttle PC. A single camera was used so the view provided by this system was not stereoscopic.

The test application was built on top of the osgART library, described in Chapter 6. Initially, it was intended to solely use the osgART's ARToolKit plugin for tracking, but it became apparent that the limitations of fiducial tracking would compromise the experiment. The particular problems were frequent marker occlusion and the need to track objects outside the field of view of the user's head-mounted camera. To remedy these problems the experiment was run within the VisionSpace visualisation centre equipped

(a) The Wiimote.

(b) The HMD with camera mounted for AR.

Figure 7.4: The Wiimote controller and HMD with tracking constellations attached.

with an ART infrared optical tracking system with sub-millimetre position and orientation accuracy (GmbH 2007). The ART tracker uses high resolution cameras to track constellations of retro-reflective spheres. This system tracked the handheld tool in the experiment while ARToolKit tracked the surface of the desk at which the user sat.

The handheld tool was a Nintendo Wiimote. The Wiimote is a wireless Bluetooth controller with a number of buttons, orientation sensors, a speaker and a vibro-tactile actuator built in (see Figure 7.4(a)). One of the ART tracking constellations was attached to the Wiimote and the cWiiMote library (Forbes 2007) was used to communicate with the device. A tracking constellation was also attached to the user's HMD so that their head movements could be recorded relative to the room (see Figure 7.4(b)). An overview of the testing environment is shown in Figure 7.5.

7.3.2 Techniques

In this section the three selection techniques are described. Figure 7.6 illustrates where these three techniques fit into the greater classification of

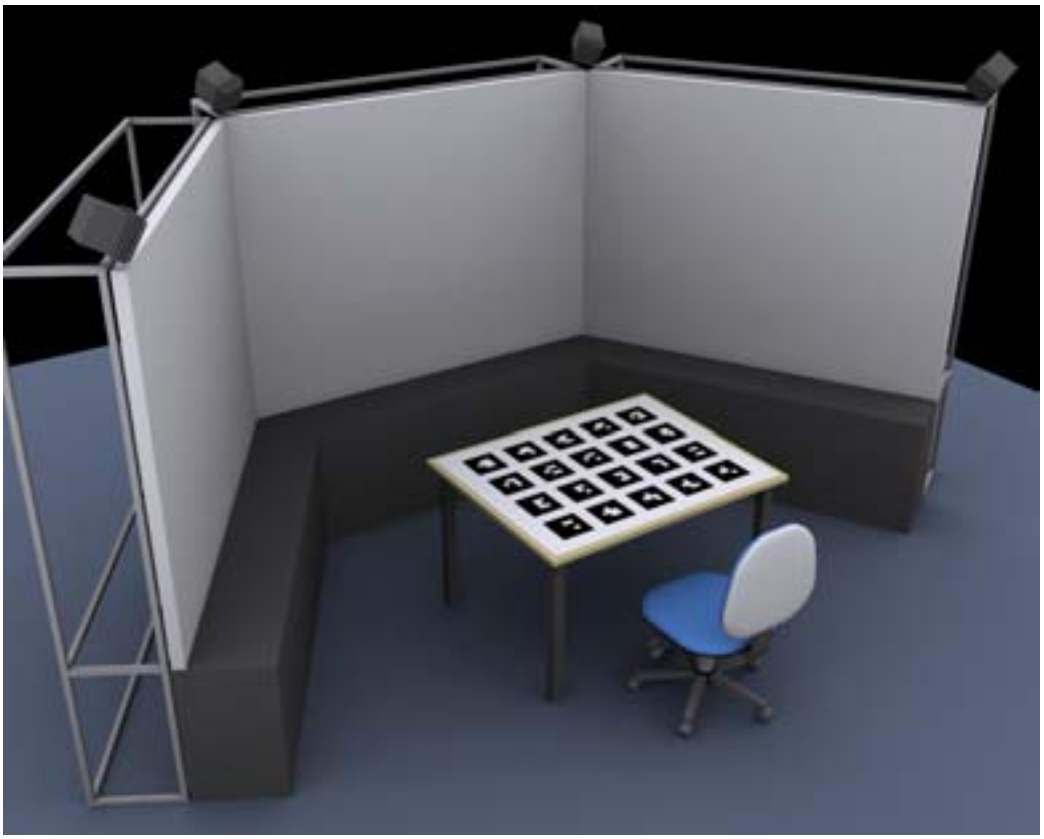


Figure 7.5: The experimental setup within the VisionSpace visualisation centre.

selection techniques. It also shows the metaphor on which these techniques are based, and a series of screenshots taken of a participant's view during this experiment.

Direct Touch

Direct Touch is a Virtual Hand technique that allows the user to select an object by simply reaching out touching it. In this interface a handheld tool was used to approximate the user's actual hand. A virtual arrow was rendered on top of the tool in the augmented reality view. There was a 1:1 mapping between the user's hand position and the position of the virtual arrow. To make a selection, the user simply needed to reach out until the tip of the arrow intersected their desired target, and pull the trigger button on the

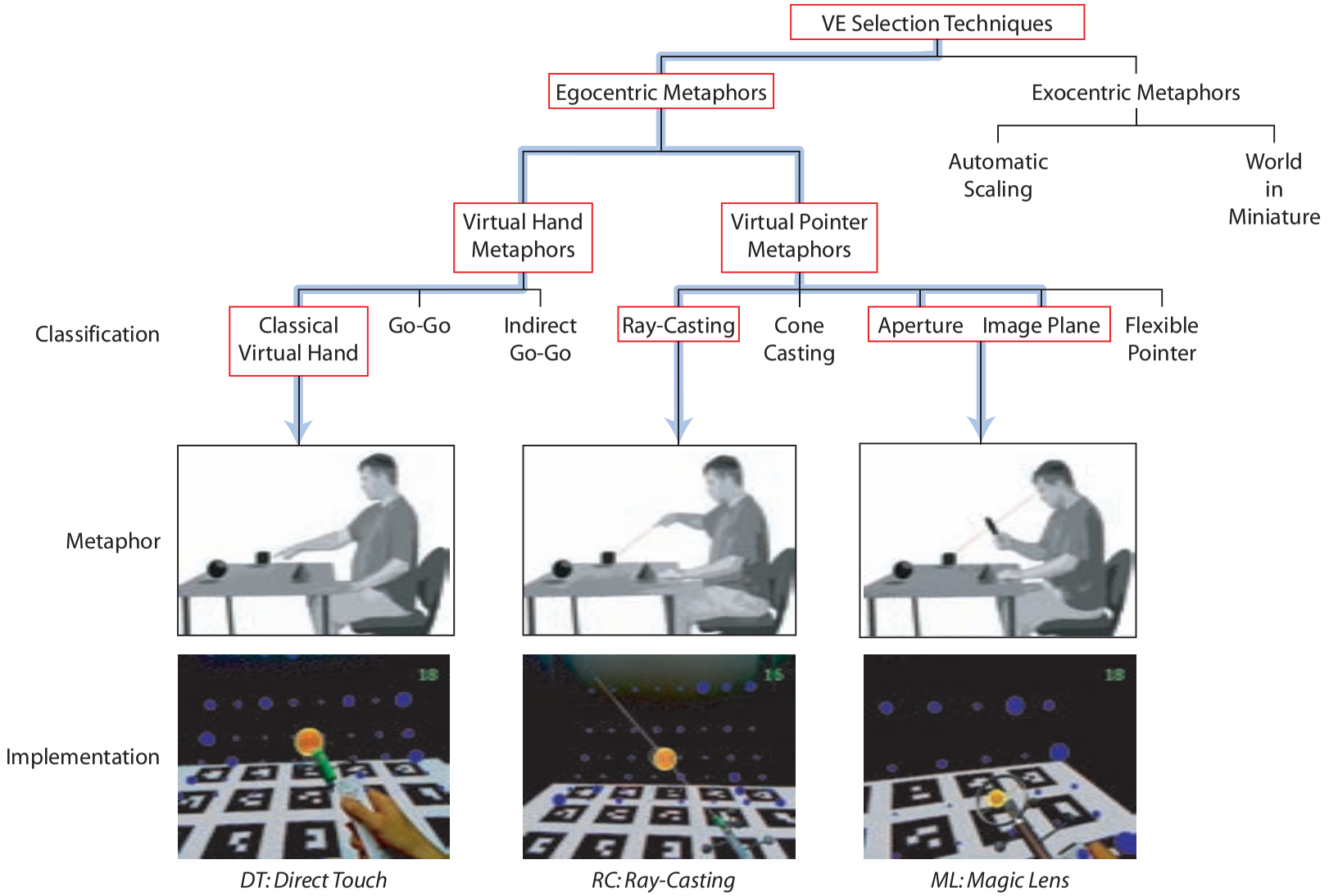


Figure 7.6: An overview of the three techniques being evaluated. Direct Touch (left) is an implementation of the classic Virtual Hand technique. Ray-Casting (middle) is the most basic Virtual Pointer technique. The Lens technique (right) is also a Virtual Pointer technique. The Lens can implement either the Aperture or Image Plane selection, although Image Plane was chosen for this experiment. All three techniques are classified as egocentric. This diagram was adapted from that of Poupyrev et al. (1998).

controller.

Ray-Casting

Ray-Casting is a Virtual Pointer technique that creates a virtual line originating at, and aligned with, the user's hand, and tests whether it intersects with objects in the scene. In this interface, the closest object out of all objects hit by the ray was selected. The ray was rendered as a thin white line extending from the controller into the scene.

Ray Casting offers several potential advantages over the other techniques. Firstly, Ray-Casting permits selection at a distance, although objects become more difficult to select the further away they are because a small change in angle at the ray's origin can move the ray selection point a great distance. Secondly, it decouples selection from the user's viewpoint so that the user can observe and select objects from different vantage points. This is not the case with the lens techniques, described next.

Lens

The Lens interface is the same configuration demonstrated in Chapter 5. That is, it is a handheld tool that looks like a magnifying glass. For this experiment, a Virtual Pointer selection technique was added to the existing AR Magic Lens interface. Image Plane and Aperture techniques, described earlier in Section 7.2.1, were considered.

Typically the view through an AR Magic Lens is altered by some effect (such as X-Ray vision). The user explores alternate views by looking through the lens as they pan it over the virtual scene. They move it nearer and farther from their eyes to reduce and enlarge the apparent size of the lens. This style of usage, as well as the lens tool's circular shape, suggest that Aperture might be an appropriate selection technique. Another option is to employ a variation of the Sticky Finger Image Plane technique, where the ray from the user's eye, passing through the centre of the lens, is used to determine object selection.

The Image Plane approach was chosen for this experiment because it does not require any physical movement of the lens in the z-axis (depth).

This was considered important because such movement is already reserved for changing the apparent size of the lens, and is also the means to adjust magnification in some applications. To support the Image Plane technique, the lens tool was enhanced with a crosshair to indicate the centre point that would be used for selection. Therefore, making selections with the lens was very much like moving a cursor over the item to be selected.

7.3.3 Participants

Sixteen participants (15 male and 1 female), ranging from ages 23 to 39, were recruited from our lab. All participants had previous experience with AR interfaces. This was a deliberate decision as we wished to investigate "expert" performance with various selection techniques and wanted to minimise the "wow factor" that routinely occurs with a participant's first encounter with augmented reality. All but one participant were right handed. The handheld controller is symmetrical and all participants used their dominant hand during the experiment.

7.3.4 Procedure

The task in the experiment was to select a single object from a set of static targets. The targets were virtual blue spheres that appeared to hang in space above the table at which the participant was sitting. They were arranged in a curved grid before the participant. This arrangement was chosen to limit the amount of occlusion between targets.

As mentioned earlier in this chapter, in 2D user interfaces Fitts' Law is a standard tool for evaluating selection performance. Fitts' Law relates human movement time to the distance and size of a target (Fitts 1954). Distance and target size are used to compute an Index of Difficulty (IoD) for each target.

In this experiment the difficulty of each task was controlled by starting with a set of known IoDs and known target locations and working backwards to calculate how large each target should be. The initial step was to select a reasonable range of IoDs. Fitts' Law models movement time based on targeting in motor space, which, in an AR interface, is the real space around

the user. Therefore, it was logical to investigate the difficulty of real world selection tasks in order to inform the choice of IoDs for the experiment.

It was ensured that the task space lay within the maximum reach of the average user (see Figure 7.7), based on the data collected by Dreyfuss (1967). The IoD values for some everyday objects were computed for two positions within the task space, shown as A and B in Figure 7.7. A is 100mm away from the starting location and B is 450mm away. The results of this informal investigation are shown in Table 7.1. From this data, it was apparent that IoDs between 1 and 7 would be suitable for the experiment. This information also serves to ground one's understanding of the difficulty of the tasks in real world, everyday terms. In the end, a range from approximately 3 to 5 was chosen. The exact values were 3.39, 3.95, 4.36, 4.67, 4.93 and 5.15.

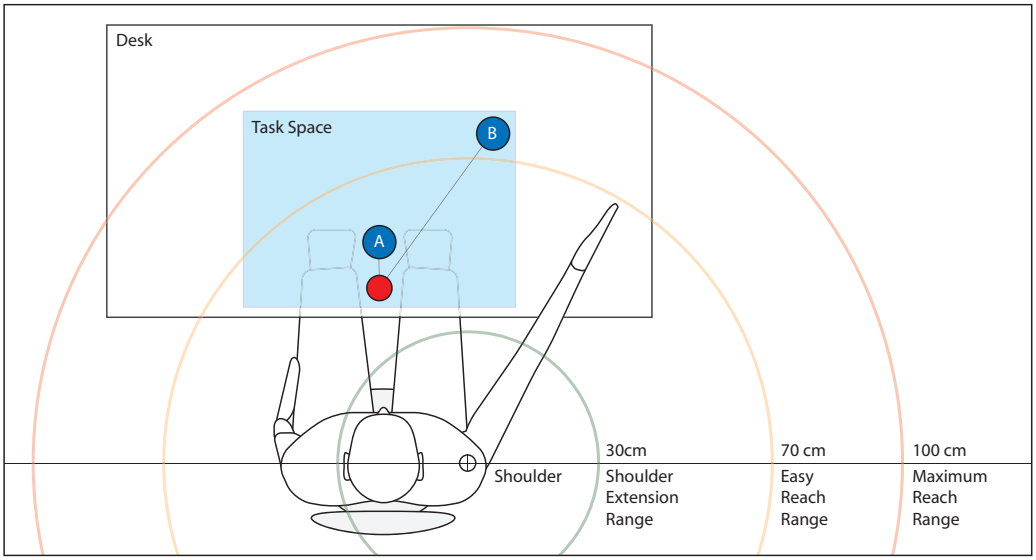


Figure 7.7: The task space was positioned within the maximum reach of the average person. Note that this diagram is two-dimensional, but the circles representing reach distance are actually spheres surrounding the user. Diagram based on the measurements provided by Dreyfuss (1967), converted to metric and rounded.

All targets were distributed spatially in front of the user, in a volume of approximately 0.5m x 0.3m x 0.5m. Each was randomly assigned one of the six IoD values and its radius was adjusted to satisfy the difficulty for that

Object	Diameter (mm)	IoD (bits)	
		At 100mm	At 450mm
Tennis Ball	63.5	1.364	3.016
Table Tennis Ball	40.0	1.807	3.615
Marble	12.5	3.170	5.209
Pea	6.3	4.077	6.178
Match Head	3.7	4.809	6.938

Table 7.1: IoD values for real objects.

IoD, as directed by the Fitts' Law equation. For example, Target X, located 30 cm away, might be assigned an IoD of 4.36 bits. In order to satisfy Fitts' Law, Target X's radius would need to be set to 7.67 mm.

In each condition of the experiment, the participant was assigned one of the three selection techniques and carried out eighteen selection tasks. Each such task began with the participant selecting a starting target. This was a red virtual sphere that always appeared in the same place: the front-centre of the virtual scene. The centre of the starting target was the point from which the distance to each target was measured for Fitts' Law calculations. Once the starting target was successfully selected, it would disappear and one of the many blue targets would turn red. The participant then had to select that target as quickly and accurately as possible. A miss caused a short error sound to play and the participant had to select again. A hit caused a success sound to play and the scene was reset to show the starting target once again, ready for the next task.

We had originally intended to provide vibro-tactile feedback through the Wiimote's internal actuator, but during initial testing participants found this feature more of a distraction than an aid so it was removed.

Before using each selection technique for the first time, participants had the technique demonstrated to them and carried out a practice round with that technique. Participants completed a short questionnaire before, during and after the experiment. An initial questionnaire collected demographic data. A questionnaire consisting of seven questions was completed for each of the six conditions. The questions, shown later in Table 7.4, were answered

on a seven-point Likert scale ranging from disagree' to agree'. A final questionnaire collected overall preference, summary information and comments. The complete set of questions are reproduced in Appendix B. Participants were compensated for their time with a $5 gift voucher.

7.3.5 Design

The experiment followed a 3x2x6 repeated measures design. The independent variables were Selection Technique ST (direct touch, ray casting, lens), Target Density TD (low, high) and Difficulty D (six levels of increasing difficulty). This design was selected to permit two types of analysis: firstly, an ANOVA to investigate differences between selection technique performance, and secondly a Fitts' Law analysis to yield a linear model of technique performance.

Participants worked through six conditions comprising all combinations of selection technique and target density. Within each condition participants carried out a block of eighteen selection tasks following the procedure described in the previous section. The eighteen tasks were divided into six sets of three, where each set contained tasks from one of six difficulty levels. All the low density conditions were done first, in a counterbalanced fashion, followed by the high density conditions, also counterbalanced. Tasks within each condition were presented in a random order.

The difference between the low and high density conditions was the number of targets. In the low density condition the grid of targets was 8x6 giving a total of 48 targets. In the high density condition the grid was 16x12 giving a total of 192 targets, four times as many as low density. The virtual scene occupied the same volume in both densities, so targets were more closely packed in the high density condition than in the low condition. The two densities and the grid layout of targets is illustrated in Figure 7.8.

The dependent variables of interest were Task Completion Time, measured in seconds, and Error Rate, measured as the number of misses made when trying to select a target. Task Completion Time was the time interval beginning when the participant selected the starting target and ending when they successfully selected the target object, regardless of how many times

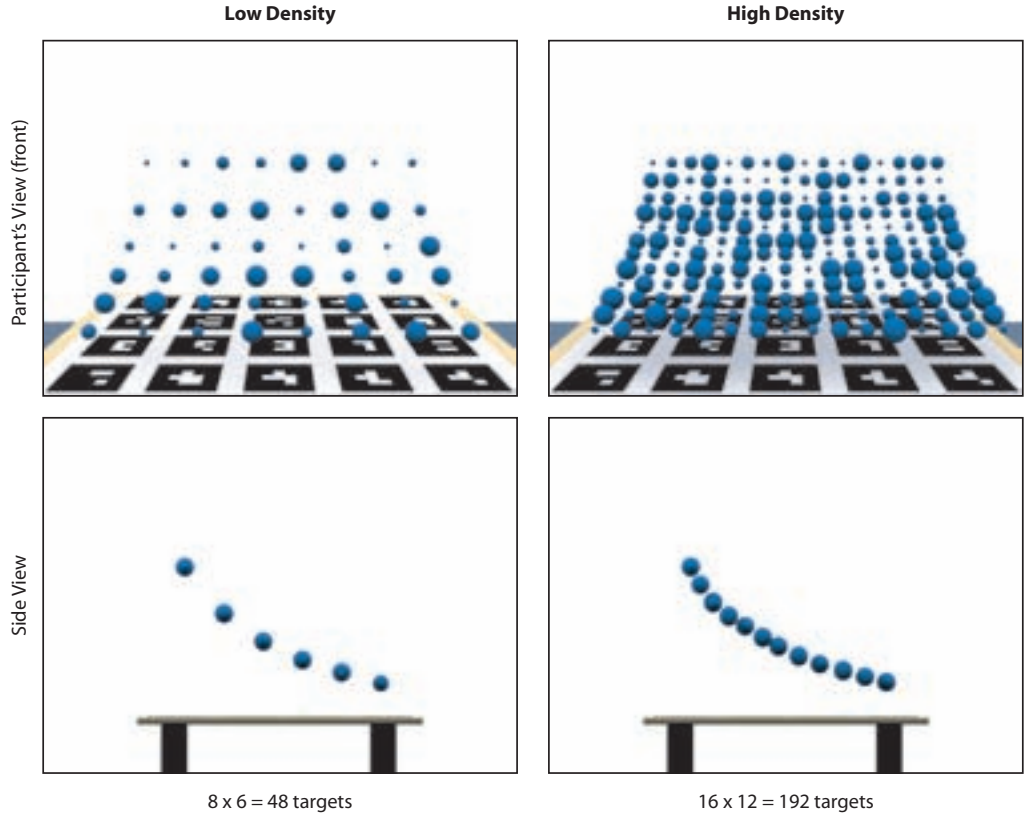


Figure 7.8: The different densities and grid arrangement of targets.

they missed. Participants were instructed to be as quick as possible but also to try to be accurate. Targets could be selected at any point on their surfaces a hit near the edge counted equally to a hit in the centre.

Other data recorded was 3D hand and head positions at intervals of approximately 10ms. This data was collected via the wide area ART tracker installed in the room. From this data, measures for average head and hand movement distance and velocity were computed.

Hypotheses

This experiment examines the performance differences between three different object selection techniques for augmented reality user interfaces. A technique's performance is typically evaluated by selection time and accuracy.

Therefore, in this experiment, the following hypotheses are proposed:

- H1: There is no significant difference in Task Completion Time between the Magic Lens selection technique and Direct Touch or Ray-Casting.

- H2: There is no significant difference in Error Rate between the Magic Lens selection technique and Direct Touch or Ray-Casting.

The techniques are evaluated across two object density levels. Object density is a known factor in determining selection performance. Therefore, it is expected that the measures of selection time and accuracy should degrade as density increases. This being the case, the rate of degradation is also of interest. A technique that degrades slowly as density increases is preferable.

7.4 Results

The sixteen participants each performed eighteen tasks in six conditions for a total of 108 trials per participant. Each trial yielded a time, miss count and 3D movement paths for the head and hand. Out of all trials, fifteen were removed because of invalid data caused by tracking failures, hardware faults (including drained batteries in the Wiimote) and participant discomfort (such as needing to adjust the HMD). The remaining data was summarised per condition. Movement paths were analysed to produce measures for average distance travelled and average velocities.

In the following sections, the results for Task Completion Time and Error Rate are presented first, followed by an analysis of the movement path data, and finally subjective feedback collected via questionnaires. All statistical analysis was performed using SPSS version 15.0. Specifically, the data was analysed using an analysis of variance (ANOVA), using Bonferroni correction ($p < .05$) for post-hoc multiple comparisons where required. In addition, the performance of each selection technique is analysed to determine whether it is accurately modelled by Fitts' Law.

7.4.1 Objective Measures

There was a significant main effect of Selection Technique on Task Completion Time ($F_{2,30} = 64.0$, $p < .05$). There was no significant effect of Target Density on Task Completion Time. Therefore, the times for low and high densities were averaged to give the overall means for each Selection Technique.

Pairwise comparison with Bonferroni correction ($p < .05$) revealed significant differences between Lens and the other two techniques, with no significant difference between Direct Touch and Ray Casting. The Lens technique performed faster than both other techniques, with a mean selection time of 3.38 seconds (*sd = 0.911*). This finding means that H1 is rejected, as there is a clear and significant difference in task completion times. The results are listed in Table 7.2 and shown in Figure 7.9.

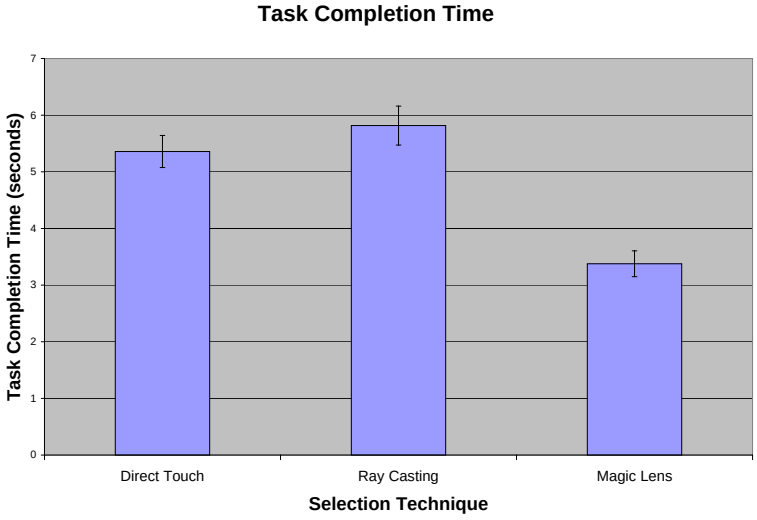


Figure 7.9: Task Completion Time for each Selection Technique. Error bars show the standard error.

Again, there was a significant main effect of Selection Technique on Error Rate ($F_{2,30} = 7.53$, $p < .05$), but no significant effect from Target Density. Figure 7.10 shows the overall mean Error Rate for each condition. Pairwise comparison with Bonferroni correction ($p < .05$) revealed significant differ-

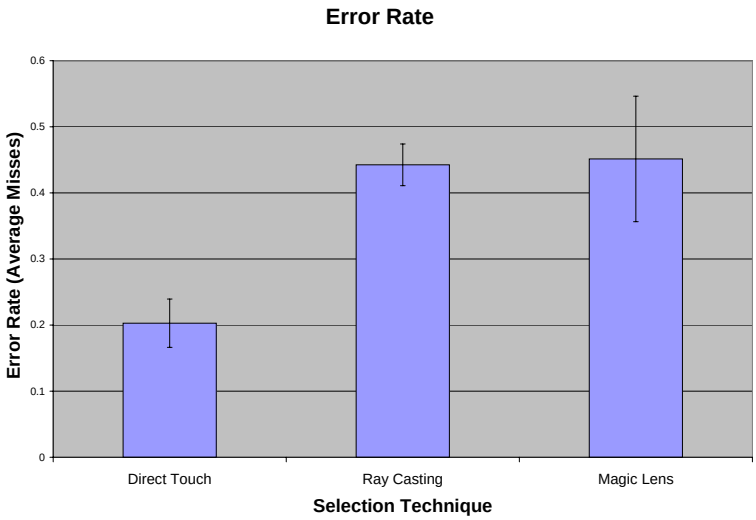


Figure 7.10: Error Rate for each Selection Technique. Error bars show the standard error.

ences between Direct Touch and the other two techniques, with no significant difference between Ray Casting and Lens. Direct Touch had a significantly lower error rate, with a mean of 0.203 misses per trial ($sd = 0.146$). These findings lead to the rejection of H2: the Magic Lens had a significantly lower error rate than Direct Touch.

Technique	Time (seconds)		Error Rate (misses)	
	mean	sd	mean	sd
DT	5.358	1.132	0.203	0.146
RC	5.816	1.376	0.442	0.126
ML	3.377	0.911	0.451	0.378

Table 7.2: Overall summary statistics for the dependent variables Task Completion Time and Error Rate.

7.4.2 Analysis of Motion

Motion data was recorded for the head and hand of each participant. This data was processed to produce metrics of Average Head Distance, Average

Technique	Head Movement				Hand Movement			
	Distance (m)		Speed (ms^{-1})		Distance (m)		Speed (ms^{-1})	
	mean	sd	mean	sd	mean	sd	mean	sd
DT	0.172	0.101	0.033	0.020	0.447	0.079	0.090	0.010
RC	0.127	0.099	0.026	0.018	0.322	0.148	0.057	0.013
ML	0.066	0.055	0.017	0.013	0.189	0.046	0.059	0.012

Table 7.3: Summary of Motion Variables.

Hand Distance, Average Head Speed and Average Hand Speed for each condition. The results of this analysis are summarised in Table 7.3 and reported below.

Movement distance results are shown in Figure 7.11. The means for Average Head Distance were significantly different for all Selection Techniques ($F_{2,30} = 21.12$, $p < .05$). Direct Touch had the longest head movement distance and Lens the shortest. Target Density had no significant effect on this metric. Selection Technique had a significant main effect on Average Hand Distance ($F_{2,30} = 23.81$, $p < .05$). All three means were significantly different, with Direct Touch taking most movement (mean = 0.172m, sd = 0.101m), and Magic Lens the least (mean = 0.066m, sd = 0.055m). Again, Target Density was not significant.

Speed results are shown in Figure 7.12. Selection Technique had a significant main effect on Average Head Speed ($F_{2,30} = 23.831$, $p < .05$). All three means were significantly different with the Lens technique exhibiting the least speed (mean = 0.017ms^{-1}, sd = 0.013ms^{-1}) and Direct Touch the most (mean = 0.033ms^{-1}, sd = 0.020ms^{-1}). Average Hand Speeds were significantly different ($F_{2,30} = 56.81$, $p < .05$). Direct Touch was significantly higher than the other two techniques (mean = 0.090ms^{-1}, sd = 0.010ms^{-1}).

Target Density did not have a significant effect on any of the motion measures. The measures were averaged across both densities and are shown in Table 7.3.

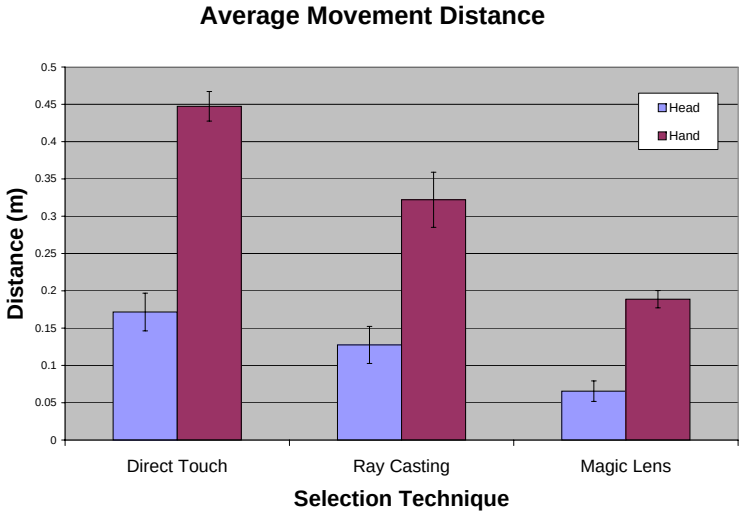


Figure 7.11: Movement distances for each Selection Technique. Error bars show the standard error.

7.4.3 Subjective Measures

A summary of the answers to questionnaire questions are shown in Table 7.4. Each question was analysed using an ANOVA, and where necessary, post-hoc analysis was performed using Bonferroni correction ($p < .05$). The results from this analysis are summarised below.

- Q1: *I found the selection technique easy to understand.* Ray-Casting and Lens showed a significant difference ($F_{2,30} = 4.34$, $p < .05$), but this is not considered to be of practical significance because all techniques were rated above 6.2 out of 7, indicating they were all well understood by participants.

- Q2: *I found it easy to select the target.* There was a significant main effect of Interface for this question ($F_{2,30} = 47.4$, $p < .05$). The Lens, rated easiest to use, was significantly higher than both Ray Casting and Direct Touch, which were not significantly different from each other.

- Q3: *I feel that I performed quickly with this technique.* There was a significant main effect of Interface for this question ($F_{2,30} = 36.4$, p

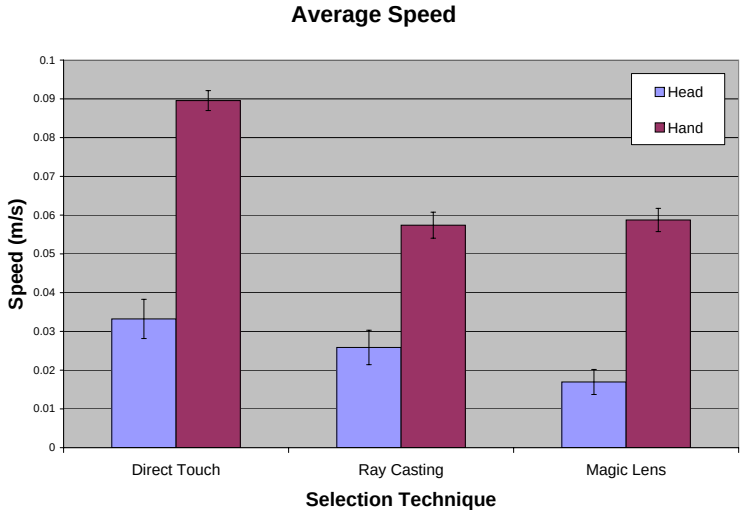


Figure 7.12: Movement speeds for each Selection Technique. Error bars show the standard error.

$< .05$). Participants felt they performed fastest with the Lens, significantly more so than with the other two techniques, which were not significantly different from each other.

- Q4: *If I had to use AR interfaces like this regularly, this is a technique I would appreciate having available.* There was a significant main effect of Interface for this question ($F_{2,30} = 43.5$, $p < .05$). Participants were enthusiastic about the Lens technique, rating it significantly higher than the other two techniques.

- Q5: *I found using this technique physically demanding.* There was a significant main effect of Interface for this question ($F_{2,30} = 24.5$, $p < .05$). The Lens technique was found least physically demanding, followed by Ray-Casting and then Direct Touch. All techniques were significantly different.

- Q6: *I found using this technique mentally demanding.* There was a significant main effect of Interface for this question ($F_{2,30} = 25.2$ $p < .05$). The Lens technique was found least mentally demanding, significantly

less so than the other two techniques, which were not significantly different from each other.

- Q7: *I found this technique frustrating.* There was a significant main effect of Interface for this question ($F_{2,30} = 34.9$ $p < .05$). The Lens technique was reported as causing the least frustration, significantly less than the other two, which were not significantly different from each other.

The Lens technique was the significantly preferred technique in Low Density, High Density and Overall (Friedman Test $\chi^2_r = 24.1$, df=2, N=16, p < .05). In each case, the next preference was Direct Touch, followed by Ray-Casting.

7.4.4 Fitts Law Analysis

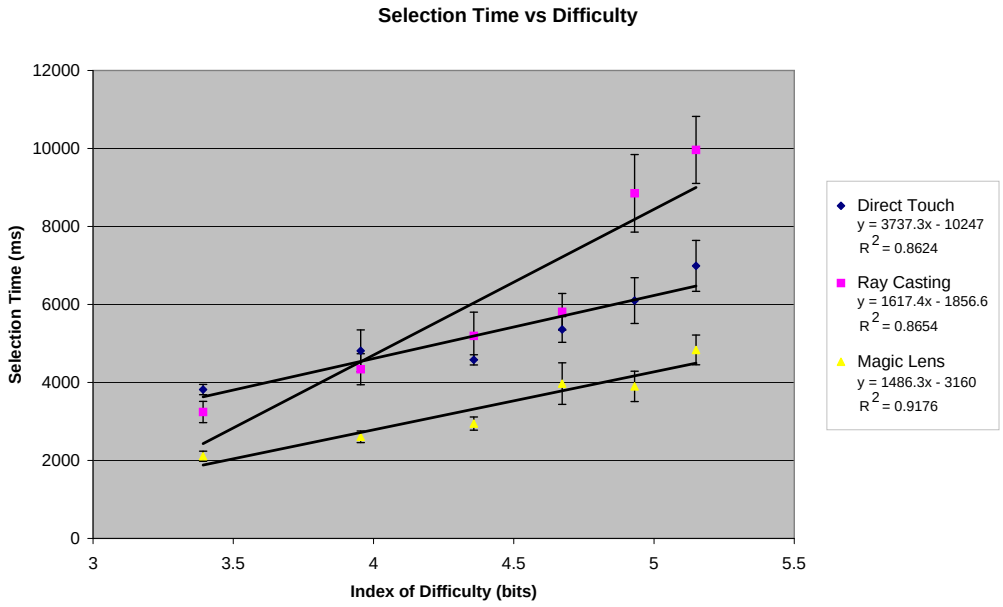


Figure 7.13: Fitts' Law Analysis.

Linear regression analysis of the relationship between selection time and IoD revealed that Fitts' Law accurately models performance with the three

Question		DT		RC		ML	
		mean	sd	mean	sd	mean	sd
1.	I found the selection technique easy to understand	6.375	0.492	6.219	0.659	6.500	0.568
2.	I found it easy to select the target	4.031	1.062	3.500	1.459	6.125	0.793
3.	I feel that I performed quickly with this technique	4.063	1.366	3.625	1.362	6.000	0.803
4.	If I had to use AR interfaces like this regularly, this is a technique I would appreciate having available	3.656	1.558	3.344	1.658	5.844	1.505
5.	I found using this technique physically demanding	5.438	1.105	4.156	1.322	3.031	1.062
6.	I found using this technique mentally demanding	4.656	1.285	5.031	1.282	2.781	1.039
7.	I found this technique frustrating	4.250	1.391	4.813	1.281	2.469	1.367

Table 7.4: Summary of questionnaire responses. Questions were posed on a seven-point Likert scale between 1 = Disagree and 7 = Agree.

selection techniques (see Figure 7.13). The r^2 values for the three techniques were 0.912 for Magic Lens, 0.862 for Ray-Casting, and 0.865 for Direct Touch. The resulting Fitts' law models are shown in Table 7.5.

Technique	Line of best fit	r^2	IoP (bits/sec)
DT	MT = 3737.3 IoD - 10247	0.862	0.268
RC	MT = 1617.4 IoD - 1856.6	0.865	0.618
ML	MT = 1486.3 IoD - 3160	0.918	0.673

Table 7.5: Linear regression equations, r^2 values, and Index of Performance for the three selection techniques.

7.5 Discussion

The Lens technique was significantly faster than both Direct Touch and Ray-Casting, leading to the rejection of the hypothesis that there would be no significant difference across techniques. It also required the least head and hand movement, followed next by Ray-Casting and then Direct Touch. These results are likely due to the fact that the experiment did not use stereoscopic AR. The lack of the stereo depth cues appears to have had the least detrimental effect on the Lens because it is based on an Image Plane technique, which reduces 3D selection to a 2D task anyway. In contrast, Direct Touch and Ray Casting suffered without stereo. The increased head and hand movements in these conditions may suggest that participants moved more to exploit other depth cues such as monocular movement parallax and overlapping to aid depth estimation. In the future, this experiment will be extended to use a stereo augmented reality system, based on the hand-held stereo visor under development at HIT Lab NZ (see Figure 7.14). The aim of that experiment will be to determine whether the lack of stereo depth cues was the only factor making the Magic Lens technique faster.

On average, participants moved their head more slowly in the Lens Technique, and moved their hand more quickly with Direct Touch. We believe the lower movement speeds and distances travelled for the Lens indicate that it requires less physical effort. This hypothesis is supported by the subjective

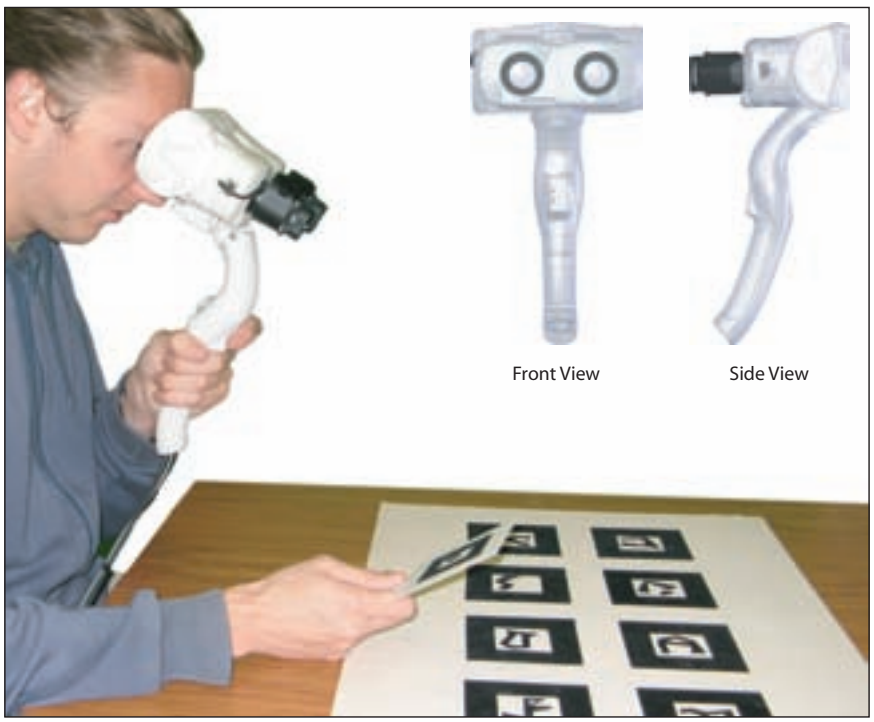


Figure 7.14: Hand-held stereo visor. This device has two cameras and a stereo-capable headset integrated within a plastic shell.

results, which strongly indicate that the Lens technique is the least physically demanding, the least mentally demanding, and overall the least frustrating of the three techniques.

Direct Touch had significantly fewest errors, leading to the rejection of the second hypothesis. The Lens and Ray-Casting had more errors, but were not significantly different from each other. One source of errors was tracking noise, which had a greater effect on the two ray-based techniques because jitter at the ray's origin can greatly affect targeting. In this experiment, the primary source of jitter was from the marker-based ARToolKit tracking, which was used for the grid off targets. Ideally, and in future evaluations, all tracking would be performed with the higher-precision ART tracker. Another source of error came from pulling the controller's trigger to make a selection, which often moved the controller enough to cause a miss - a situation that participants found most frustrating.

Interestingly, target density had no significant effect on selection time, error rate, movement speed or movement distance. This is counter-intuitive, as logically performance should degrade as density increases. This may suggest that the choices for density levels were not extreme enough, or that there is some other factor at work that overshadows the differences between densities. For example, because the selection techniques used in this experiment required precise pointing, they may be immune to increases in object density. Techniques with "soft selection" such as cone-casting or aperture are likely to be affected to a greater degree.

7.6 Future Work

There are several factors that could be considered in future evaluations of lens selection techniques. The use of stereoscopic cameras and displays will make many selection techniques more viable. Many participants complained about the lack of depth perception hindering their performance, especially in the Direct Touch conditions. It could be argued that the lack of stereo vision in this experiment biased it towards the Lens technique, however this argument only highlights the fact that since most current tabletop AR setups are not stereoscopic, selection and manipulation techniques that rely on reasonable depth perception should be avoided.

In this experiment, the chance of targets occluding each other was intentionally minimised. This is obviously not the typical scenario in many 3D virtual environments and future work will need to address how these techniques cope under increased object density and occlusion.

This experiment also only tested single object selection. Multiple object selection is often required, and is a cause of frustration if the user is forced to make many single selections rather than using an appropriate multiple object selection technique. In terms of virtual lens interfaces, this could be a reason to investigate Aperture selection rather than the Image Plane technique used in this experiment.

7.7 Summary

This chapter reports on a formal evaluation of selection techniques for tabletop augmented reality user interfaces. Object selection is a crucial task in user interfaces because it precedes almost all other tasks the user might wish to perform. The three selection techniques compared in this study were Direct Touch, Ray-Casting, and Lens: an Image Plane technique intended for use with virtual Magic Lenses described in Section 5.4.

The Lens technique was found to be faster than the other techniques and although it was not the most accurate, it required less head and hand movement, and lower head movement speed. This suggests that it is a more enjoyable technique, which is supported by participant feedback that shows the Lens technique was least physically demanding, least mentally demanding, least frustrating, and preferred in all cases.

Target Density was found to have no significant effect on any of the measures analysed in this experiment. Further testing is required to determine whether the density levels were simply not different enough, or whether some other factor in the tabletop AR setup hinders selection performance by some constant factor, such as a large preparation cost, that outweighs the effect of density.

Chapter 8

Flexible Sheet Lenses

In this chapter, a new a new variation of the Tangible AR Magic Lens concept is explored by experimenting with flexible and resizeable lens shapes. Inspiration for this new type of lens came from a number of sources, particularly a scene from the movie Red Planet[1] in which a flexible sheet is used as an interface device (see Figure 8.1(a)), as well as the recent advances in digital paper and flexible OLED (Organic Light Emitting Diode) displays.

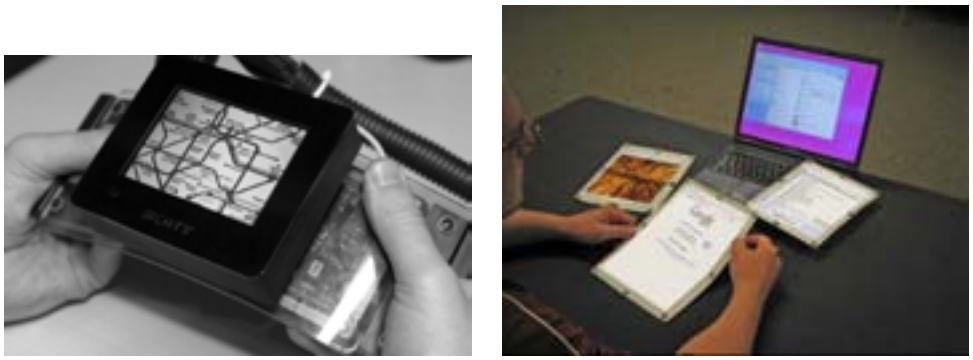
The new lens is illustrated in Figure 8.1(b). It can be described by two physical handles attached to a flexible visualisation surface. The surface changes dynamically in shape and size and is defined implicitly by the location of the two handles. Furthermore, the lens can also be used with a single handle, described later, providing all the standard techniques proposed in previous chapters on AR Magic Lens interfaces.

(a) A Flexible Lens-like Information and Navigation interface from the movie *Red Planet*.

(b) The Flexible Tangible AR Magic Lens.

Figure 8.1: Flexible Magic Lenses as both physical surfaces (left) and virtual surfaces (right).

[1] *Red Planet*, 2000. Directed by Antony Hoffman. http://redplanetmovie.warnerbros.com



(a) The Gummi bendable computer of Schwesig et al. (2004). Image courtesy of Carsten Schwesig.

(b) PaperWindows of Holman et al. (2005). Image courtesy of Roel Vertegaal.

Figure 8.2: Flexible display technologies.

This new type of lens provides several advantages over previous types. This approach offers a new way to manipulate and explore data by exchanging a rigid physical lens shape for a virtual non-rigid surface. Freed from physical constraints, the lens can be naturally manipulated and positioned into locations that might be difficult to access with a physical tool. For example, you cannot place a physical tool inside another solid physical object, but by holding two handles on either side of a solid object, the virtual lens surface can still pass in between.

Furthermore, by introducing bimanual interaction to the system, there are possibilities of using gesture interaction to manipulate the parameters and functionalities of the lens. In this fashion, this approach follows in the footsteps of recent works like the Gummi bendable computer (Schwesig, Poupyrev and Mori 2004) or Paper Windows (Holman, Vertegaal, Altosaar and Johns 2005) that explore flexible surfaces for interaction.

Finally, since the lens is not a physical surface, it need not occlude objects behind it. With other implementations where a physical lens prop is used as the tangible element, the real world behind the lens is hidden. The virtual lens surface allows us to create complex effects such as applying image processing operations to real objects through the lens.

(a) Single Handle Mode. (b) Dual Handle Mode.

Figure 8.3: Different flexible lens configurations.

8.1 Interaction Techniques

The tangible and flexible Magic Lens offers a large range of interaction techniques, mainly based on the physical manipulation of the handles and deformation of the lens surface. The lens can be used in two modes: *single handle* or *dual handle*. Both of these modes give access to a multitude of functionalities supported by these new techniques. In this section these two interaction styles are described.

8.1.1 Single Handle Mode

In the single handle configuration the lens can be used like a standard AR Magic Lens (see Figure 8.3(a)). In this mode it provides the means to filter different types of data, supporting similar functionalities as the techniques presented earlier in this thesis, as well as additional techniques like those of Schmalstieg, Encarnaçao and Szalavári (1999) in their work on props for the Virtual Table.

The lens surface is defined by a rectangle attached on the side of the handle. However, in contrast to previous works, the user can dynamically resize the width and height of the lens and access different modes of filtering by manipulating physical controls present on the handle.

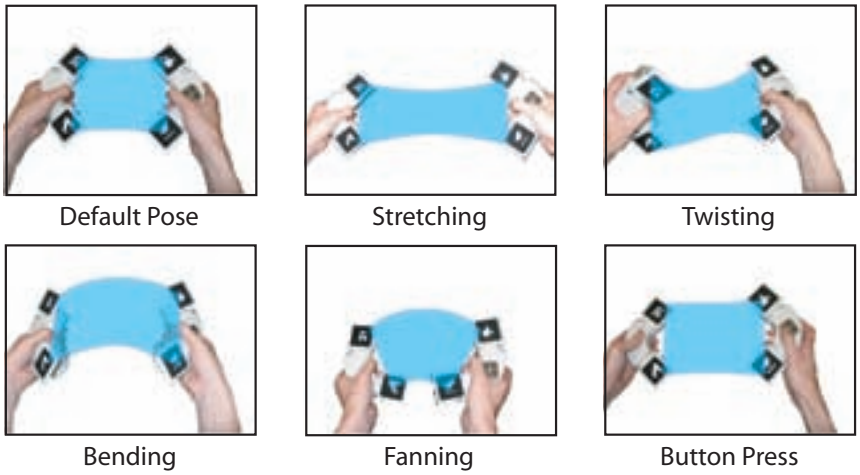


Figure 8.4: The set of lens gestures explored so far.

8.1.2 Dual Handle Mode

In the dual handle mode shown in Figure 8.3(b), the user holds one handle in each hand to manipulate the size and curvature of the Magic Lens sheet. The lens is a curved surface parameterised by the position and orientation of the handles. The width of the surface is controlled by the distance between the two handles. The height can also be adjusted but is typically fixed to the length of the handles.

Supplementary functions can be supported by the accessible range of different bimanual postures that can be realised with the lens. In this work, efforts are limited to a default pose and four additional poses based on *stretching*, *fanning*, *bending*, and *twisting* (see Figure 8.4). These poses mimic the effect of manipulating the lens surface as if it was a physical, pliable surface. The pose recognition is performed by a neural network technique that is described further in Section 8.2.2.

The *stretching* pose is formed by moving the handles further apart along the x axis, resulting in the horizontal expansion of the lens. Opposite rotations around the y axis of each handle corresponds to a *fanning* gesture (the lens splays out like a Chinese fan). *Bending* is the inward and outward rotations of the handles around z. Finally, opposite rotations of the wrists around the x axis describes the *twisting* gesture. When the handles are held

in the same plane at a reasonable distance apart it is in the default pose.

Clearly some gestures will be of more practical use than others. For example, the twisting gesture results in much of the lens surface being hidden from the user, as it tends towards being perpendicular to the user's view. However, there is still potential for such gestures as one-off motions to activate functions in the user interface. A quick twist of the lens, for example, could move forward through a set of data layers, or toggle descriptive labels.

In addition to determining which pose is currently held, a collection of metrics are also continually updated. These metrics can be used in conjunction with the current pose to parameterise any actions resulting from the pose. For example, once the stretching pose has been activated, the metric for the distance between the handles could be used to adjust the zoom level of the lens view.

A technique has additionally been designed to easily switch between single and dual handle modes. When the user is in single handle mode they can attach the other handle to the virtual surface of the lens. The collision between the handle and the surface is detected and the mode is switched to dual handles mode. To switch back, the user swiftly bends one of the handles until it makes an orthogonal angle with the other handle, "snapping" the lens apart.

8.1.3 Lens Functions

The lens can be used by default as a visual filter on 3D data. In addition, the gestures provide a new level of interaction to analyse and manipulate the data. They support exploration of the data by allowing the user to quickly and interactively set the values of the parameters of the visual filter. For example, by simply bending the lens the user could cycle through various geographic layers overlaid on a virtual globe. The strength of the bend (determined by a metric based on the angle between the normals of the handles) could be mapped to the speed at which the layers of data are scrolled through. Another example, shown in Figure 8.5, uses the fanning gesture to adjust the zoom level of the scene.

Another category of tools made possible with the flexible lens is those

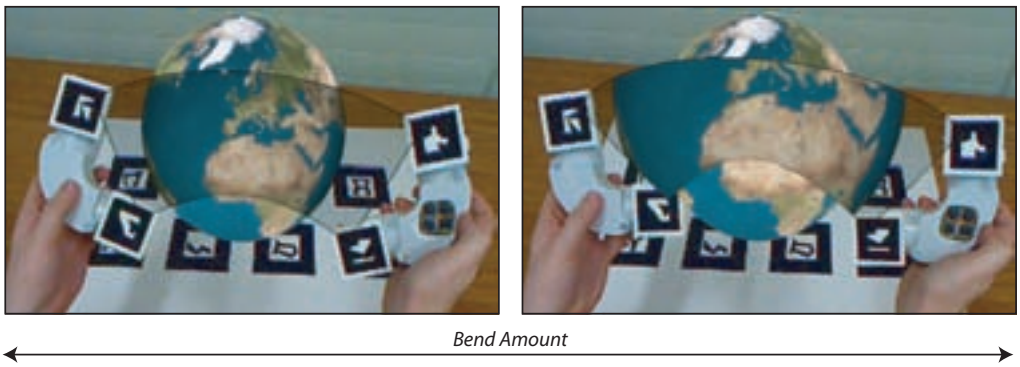


Figure 8.5: The user can zoom in on the virtual globe by using a fan gesture with the lens.

that use the surface like an interactive 2D canvas. This idea mimics the concept of the see-through tools demonstrated in the ToolGlass interface of Bier et al. (1993). The surface can be used to present additional information about the visual filter and its parameters, the current pose the lens is held in, and application specific information related to the scene.

Using the direction of the user's view a cursor can be placed on the lens surface to further enhance its 2D interface capabilities. Conversely, the positions of elements in the 3D augmented reality scene can be projected back onto the lens surface and incorporated as well. For example, when viewing a 3D terrain model, the important locations could appear as labeled points on the lens surface along with any extra relevant information. Some possible applications of the 2D interface capabilities of the lens are shown in Figure 8.6.

8.2 Implementation

Physical Handles and Tracking

The flexible Magic Lens was implemented using the osgART library described in Chapter 6. osgART's ARToolKit plugin was used to track the markers on the handles. To increase the stability of the tracking each handle had two markers attached. By tracking two separate points we rely more on the

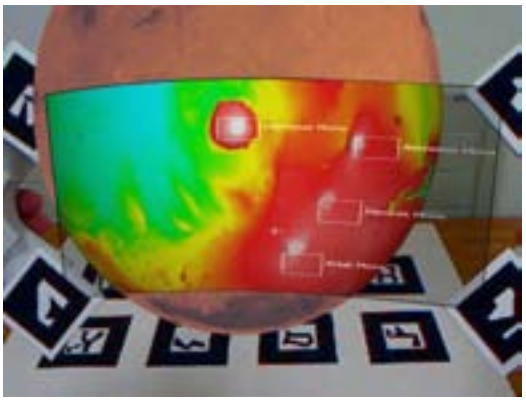


Figure 8.6: 2D interface components embedded in the lens surface. The crosshair is a cursor positioned using the intersection of the user's gaze vector with the lens. The four icons are positioned by the projections of four landmarks within the scene.

position tracking of ARToolKit rather than the orientation tracking, which is the most unreliable.

The two handles, shown in Figure 8.7, were constructed from plastic tubing and aluminium. A small wireless radio controller was embedded within the right handle. This controller provided four button inputs arranged in a standard North-South-East-West configuration. The buttons were operated by the user's thumb. The state of the buttons was transmitted to the computer and could be used for any of a wide range of interactions. The wireless controller and receiver were built by the HIT Lab NZ's electrical engineer, Dr Marilyn Lim.

The wireless controller was added for two reasons. Firstly, simply to provide additional input capabilities to the user. Secondly, a potential user interface problem was identified and the buttons provided a solution. As an example, the user may wish to increase the viewable area of the lens by moving the handles apart, and in doing so, inadvertently perform a stretching gesture. A solution to this type of problem is to only monitor for gestures when one of the buttons is held down, allowing the user to move the handles freely for the remainder of the time.

We also experimented with removing the handles completely and tracking

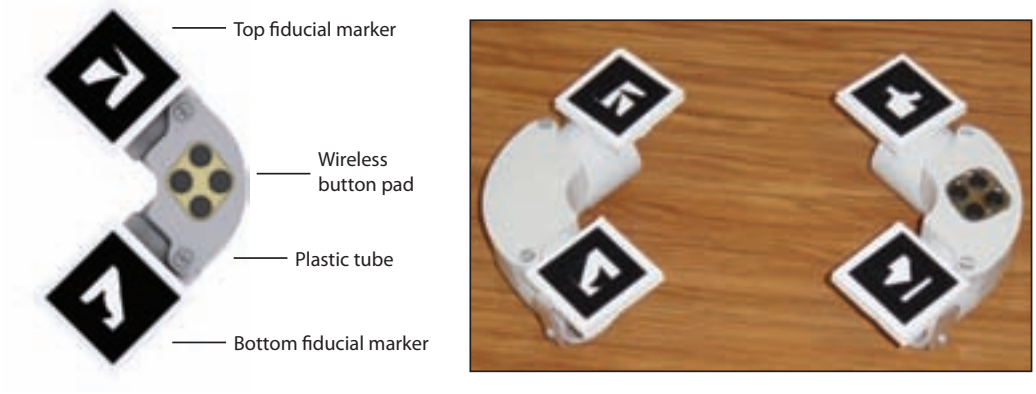


Figure 8.7: Flexible Lens Handles.

(a) Finger Tracking

(b) The Tinmith User Interface. Image courtesy of Wayne Piekarski.

Figure 8.8: Finger Tracking.

fingers instead, as shown in Figure 8.8(a). Tracked fingertip interaction for augmented reality interfaces has already been explored in previous work, such as the Tinmith system of Piekarski and Thomas (2002) shown in Figure 8.8(b), the Magic Rings of Dias, Barata, Santos, Correia, Nande and Bastos (2003) and the FingARTips system of Buchmann, Violich, Billinghurst and Cockburn (2004).

8.2.1 Lens Rendering

In single handle mode the shape of the lens is simply a flat sheet. In dual handle mode, the surface is generated by interpolating between the control points defined by the four markers (two on each handle). The surface is a grid where the start and end points of each row are calculated by linear interpolation between the two markers on each handle. The remaining vertices in each row are calculated using Hermite interpolation to produce an attractive flowing surface. The number of rows and columns in the grid can be adjusted to increase smoothness. The calculations to produce the mesh seem relatively inexpensive: an overly detailed mesh of 30x30 cells easily runs in real-time on a modern computer with a 3D graphics accelerator card.

The rendering process of the flexible Magic Lens is illustrated in Figure 8.9, and follows these steps:

1. Use tracker to determine the 3D positions, normals and relative distances of the four markers.

2. Tessellate the surface by interpolating vertices and normals between the four points.

3. Render the lens view into a dynamic texture and apply this texture to the surface using projective texture coordinates.

4. Render the context view in the standard way (or in the case of real objects comprising the context, do not render anything outside the lens).

5. Repeat...

Using this technique, many different types of virtual content can be overlaid on other virtual or real objects. The virtual content can be scientific data such as volumetric datasets, nested data like the internal structures of a building, or other visualisations where we can manipulate the visualisation parameters.

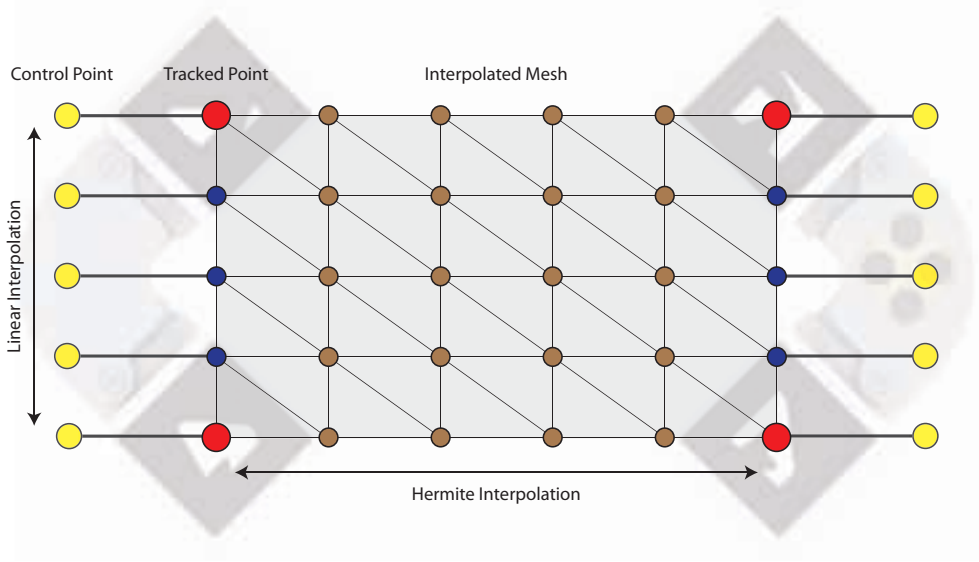


Figure 8.9: The rendering process of the Flexible Magic Lens surface. The red dots are tracked points (markers). The yellow dots are control points determined by fixed offsets from the tracked points. Blue dots are linearly interpolated points to create the beginning and end points of rows. Brown dots form the surface of the sheet.

8.2.2 Interaction

To implement the pose recognition we use an Artificial Neural Network (ANN). An ANN is an appropriate approach in this case because it allows a number of arbitrary poses to be trained without requiring them to be defined and differentiated programmatically (as long as all classes of pose are linearly separable). Many different poses can be trained and recognised accurately. A pose is characterised by the positions of three of the tracked markers relative to the fourth. The parameters that define the pose are the heading, normal vector and distance of each marker. This gives an array of seven values for each of the three input markers, making a total of 21 values per pose.

The neural network was trained by positioning the handles in an arrangement we wished to map to a certain pose. The 21 values defining that arrangement are constantly calculated in realtime. By pressing a key, the current set of values, along with the chosen pose, are incorporated as a training example into the neural network. The network can be retrained immediately with all available examples. This allows easy tuning of the neural network during runtime. Furthermore, it is possible to train different databases of examples for different situations, such as for different users or different task types. For example, a small set of poses may support a sufficient number of actions for simply browsing through a dataset, but a much larger set of poses may be needed to support query, manipulation and analysis actions on that data.

We have currently trained the system to recognise five different gestures. We chose the open-source library `fann`[2] to implement the ANN. Nissen (2003) describes the development of `fann`.

The steps taken during the pose recognition phase are listed below and illustrated in Figure 8.10.

1. Interactively train the ANN by arranging poses.

2. Extract parameters of tracked points and feed them into a trained ANN.

[2] As of September 2007, `fann` can be downloaded from `http://fann.sourceforge.net/`

3. Use the result of the ANN to apply interaction and generate modified view.

4. Repeat from step 2...

One of the handles is equipped with a small wireless button pad. The pad uses a radio connection to communicate with the computer. The pad is mount symmetrically so that the handle can be flipped for left-handed users. The four buttons on the pad can be programmed to add more control functionalities to the lens in a similar way to Brown and Hua (2006). Button usage can be combined with the pose recognition to provide a greater range of user interface actions.

8.3 Applications

In this section we describe a small set of sample applications we have developed to demonstrate the possibilities of the Flexible Magic Lens.

8.3.1 Geographical Information Systems

We have explored the usage of the lens in the context of geographical information systems (GIS). We developed a small application showing a 3D virtual globe in a desktop AR system. The lens is used to explore different types of data, such as temperature, population density and landcover. This exploration is done by using a *bending* pose to switch between the different layers (see Figure 8.11). Once the bending pose is detected, the metric for the bend amount is used to select which layer to show, giving the user rapid access to each layer. Since the lens can be resized at anytime by moving the handles, the user can choose to enlarge or reduce the focus area, giving them flexibility during observations and analysis.

Fatigue can rapidly set in when trying to hold the lens steady in the air. Therefore, we enhanced the system with a mechanism that permits the user to detach the lens surface from the handles, providing a way to "copy" the virtual lens. The user can "stamp" lenses at various positions in the scene, creating a gallery of alternative viewing portals into the data (see Figure

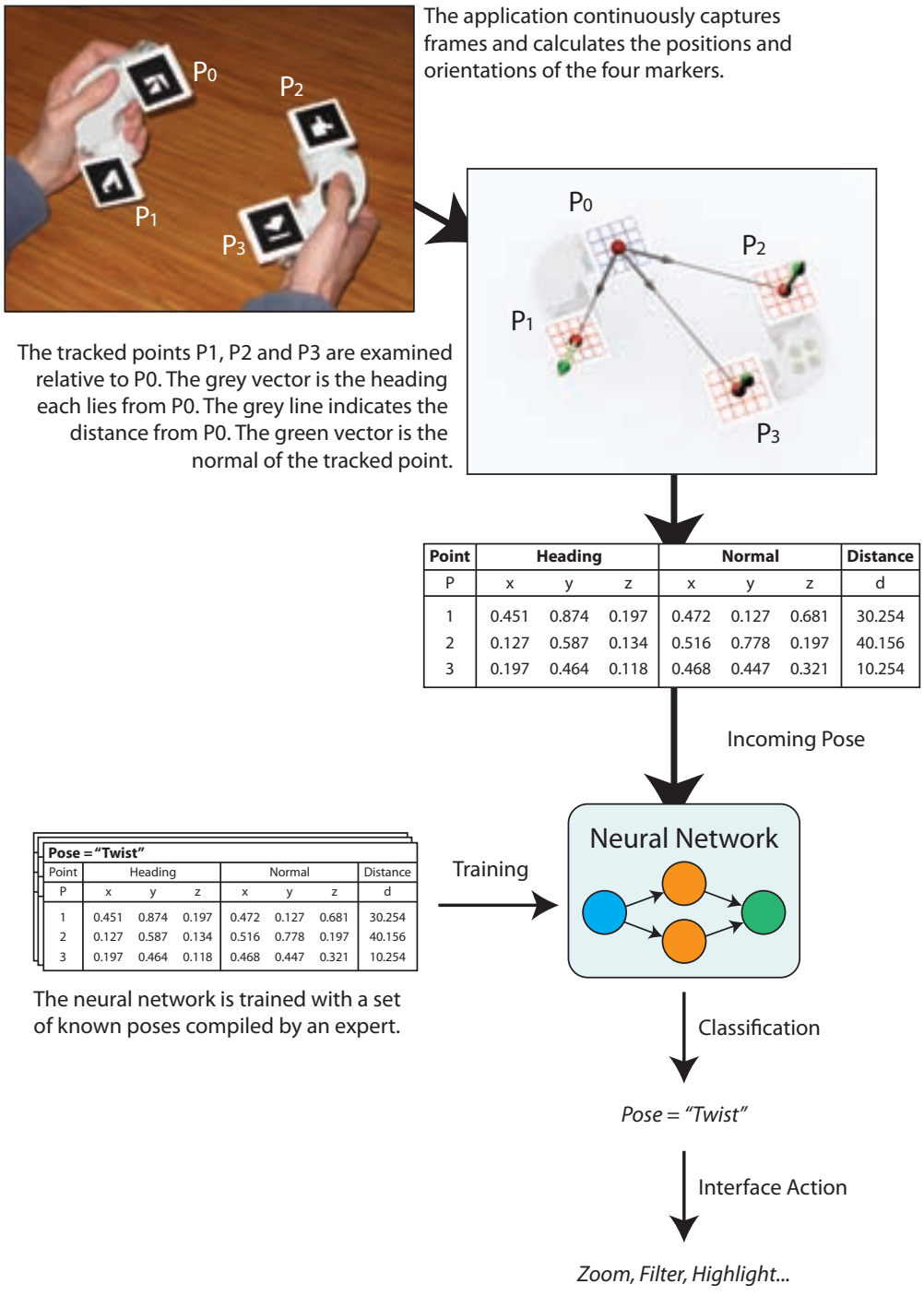


Figure 8.10: The pose recognition process of the Flexible Magic Lens surface.

Figure 8.11: Different layers of information can be accessed by bending the lens surface.

8.12). We are currently exploring efficient ways to manage scenes containing multiple lenses.

We have also explored how our system can be used for multidimensional and multivariate data like that found in atmospheric research. In a collaboration with the Centre for Atmospheric Research (UCAR) at the University of Canterbury, we are developing tools to enhance the visualisation of complex weather simulation models. In a first implementation, we developed a tool that allows the user to browse through the steps in a weather simulation by *stretching* the lens.

8.3.2 Engineering

The lens can be used to visualise the hidden internal structures of real objects. This solution can be valuable in the context of education and engineering. For example, Figure 8.13 demonstrates the usage of the lens for observing the stack of complex layers that make up the panel inside an LCD screen. Rather than having a complicated interface to change the visualisation of this component or using a simple desktop application, the user can simply place the lens over the screen and perform a *stretching* gesture to see an exploded, annotated view of the components.

Figure 8.12: Multiple Lenses stamped around a globe, each showing a different source of information.

8.4 User Feedback

We have conducted an explorative study of Flexible Magic Lenses based on our prototype designs. Our intention was to gather feedback from users in terms of the usability and enjoyability of the flexible lens. We asked six volunteers to try the lens in a short pilot study. We chose people familiar with AR technology because we were specifically interested in their opinions of the flexible lens concept, rather than AR in general, which has a high novelty factor amongst first time users.

In the study the participants were given a brief introduction to the system, and then asked to explore the different interaction techniques provided by the interfaces. Observations of participants were noted during the evaluation and informal interviews at the end were used to collect additional information. Three interaction scenarios were presented, based on the demonstration applications described in the previous section: a static cutaway view of an electrical appliance, an interactive cutaway view of an LCD monitor and a visualisation of data layers on a virtual globe.

In the first scenario, participants could use the lens to see inside an electrical appliance (see Figure 8.14). Visualisation was the only feature supported (there were no additional actions mapped to lens poses or buttons). Although participants liked the flowing surface of the dual handled lens, some

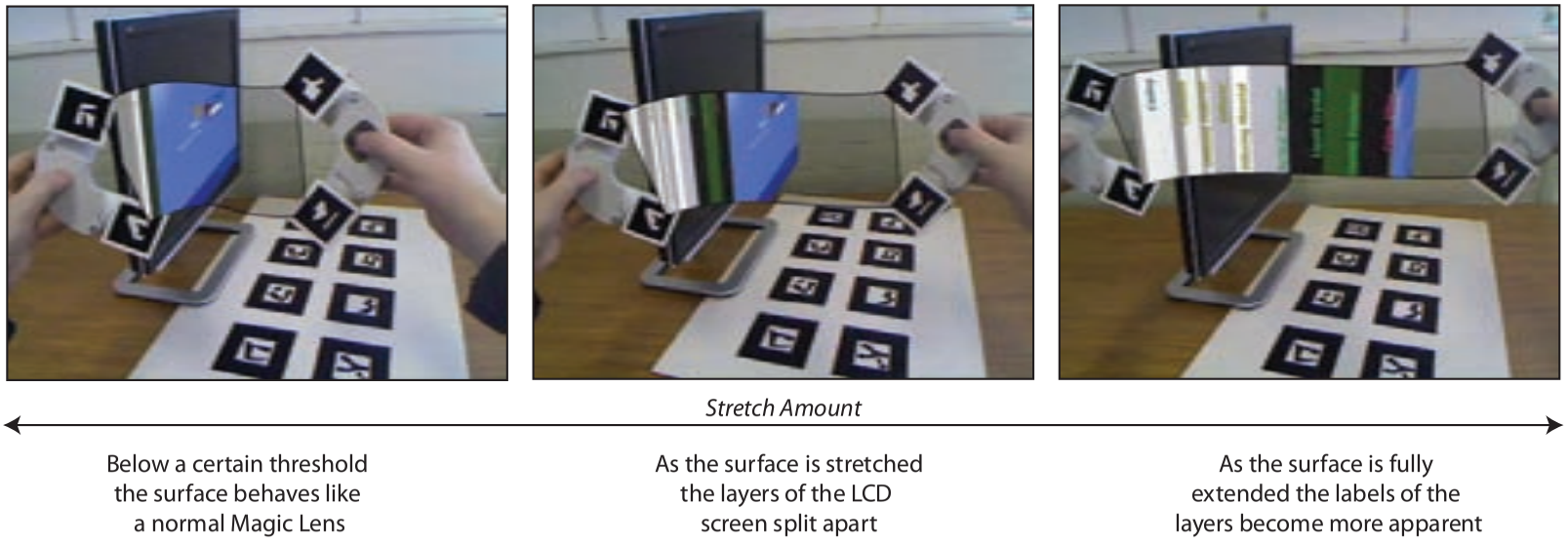


Figure 8.13: A flexible lens showing the layered components of an LCD monitor. The layers expand apart as the lens is stretched.

Figure 8.14: Looking inside an electrical appliance with the flexible Magic Lens.

quite rightly pointed out that two hands were not necessary in this case, and therefore preferred the single handled lens.

The second scenario showed the internal layers that comprise an LCD panel. This 3D model was seen through the lens, overlaid on a real LCD monitor (see Figure 8.15). When the lens was stretched, the layers moved apart so that each could be separately identified. Labels for each layer also appeared and grew larger based on the degree of stretching. All the participants stated how naturally the stretching motion mapped to the expanding layer view.

The third scenario presented a virtual globe as shown in Figure 8.3. Different layers of information could be textured onto the globe when seen through the lens. Five such layers were loaded and the degree to which the lens was bent determined which layer to apply. By bending the lens more or less, the entire set of layers could be traversed. Participants did not find this mapping as logical or comfortable as that of the second scenario. The main problem was that in order to remain on a particular layer, the user had to hold the lens still at a certain bent position. Another problem was that there was no feedback as to when the layer would change. Through discussions it was hypothesised that for distinct layers (such as population density and pollution level) it would be better to move between layers with a well-defined action such as a quick bend forward and then back again. On

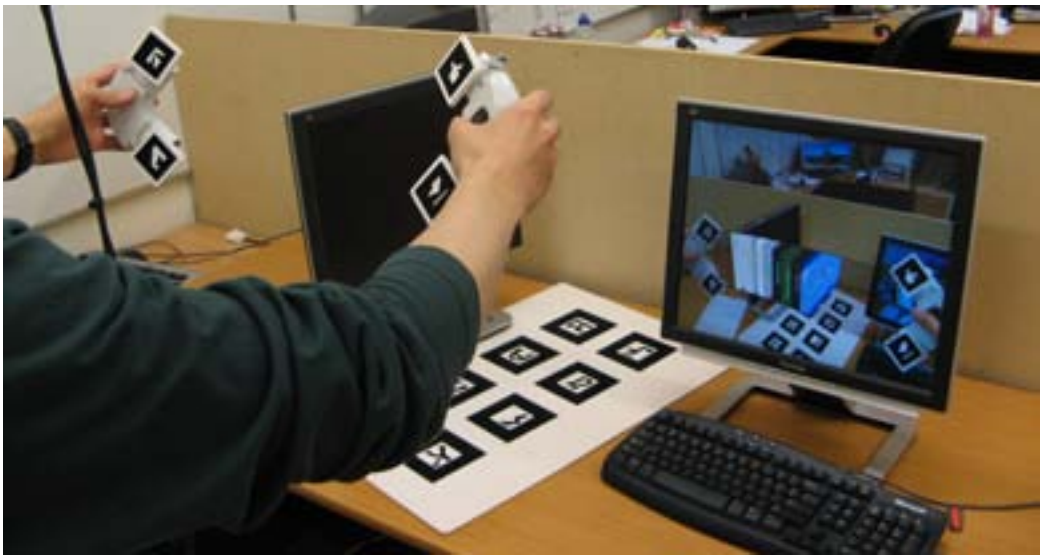


Figure 8.15: The flexible lens being used to look at the layers of an LCD screen. Photo courtesy of Andreas Dünser.

the other hand, for continuous data (such as weather features sampled every day for a month) the current technique of gradually scrolling through the collection could be appropriate.

In general the feedback was positive. Participants learnt how to use the flexible lens very quickly. The largest problem was loss of tracking due to marker occlusion. The small markers on the handles often moved outside the camera's field of view, or covered up the markers on the large table-top tracking grid.

Several participants commented that their arms got tired quite quickly when using the lens handles. The current handles are crude prototypes and could be substantially lighter and more ergonomic. One participant was particularly pleased with the stamping mechanism which fixes lenses in place.

The major finding emerging from this short evaluation was that suitable mappings between gestures and interface actions are vital, and tuning those mappings to make them as intuitive as possible is important. We are encouraged on this point with the LCD panel application used in the second scenario, but clearly the virtual globe's use of bending requires further thought.

8.5 Summary

In this chapter, we have reported on the development of the Flexible Sheet Lens, a variation of the AR Magic Lens introduced in Chapter 5. The flexible lens is a resizable visualisation surface controlled by two physical handles. The surface is suspended between the handles, allowing for easier augmentation of real world objects compared to the magnifying glass type.

Bimanual interaction and pose recognition increase the interactivity of the system. Pose recognition is achieved using an Artificial Neural Network. Although the ANN performed well in informal tests, a method to evaluate the accuracy of the pose recognition is required. The flexible sheet lens provides a highly interactive user experience. The next challenge is determining suitable mappings between the possible interactions and interface events.

We have begun to explore the development of flexible volume lenses. The volume mesh is generated by simply repeating the procedure for the flexible sheet four times, once for each side of the volume. One benefit of the volume version of the flexible lens is that it is view-independent and could therefore have more utility in collaborative Augmented Reality environments.

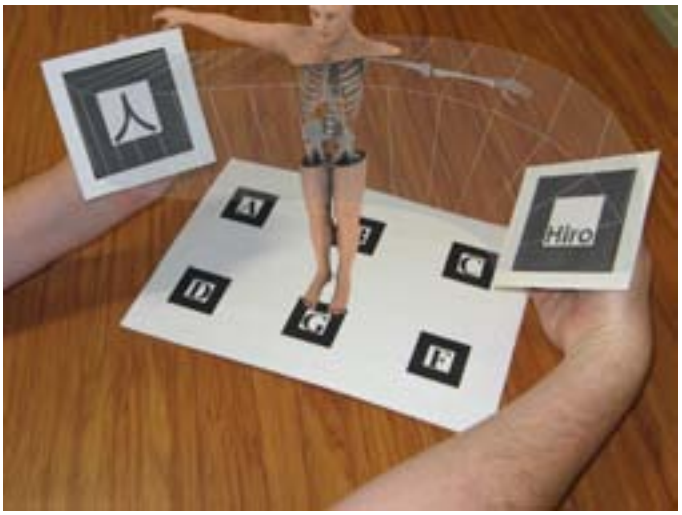


Figure 8.16: The Flexible Volume Lens is an extension of the flexible sheet lens. A 3D volume has the advantage of being view-independent, in contrast with a sheet, which produces different views depending on the location and orientation or the viewer.

Chapter 9

Discussion and Future Work

9.1 Discussion of 2D Magic Lenses

Technology flows in waves and currently multi-touch displays are gaining popularity, from small-screen devices such as Apple's iPhone, to Microsoft's Surface table-computing platform, to wall-sized, multi-user systems such as the impressive demonstrations of Jeff Han[1]. Figure 9.1 illustrates these innovations.

(a) Apple iPhone. (b) Microsoft Surface. (c) Multi-touch screen of Jeff Han.

Figure 9.1: Multi-touch screen technologies.

Magic Lenses have great potential as user interface tools on multi-touch screens, and have already appeared in some demonstrations. The manipulation cost encountered in the 2D Magic Lens experiment reported in Chapter 4 may be almost completely eliminated when mouse input is replaced by natural two-handed gestural input.

NextWindow is a company that develops touch sensitive screens based on

[1] Jeff Han, Perspective Pixel, `http://www.perceptivepixel.com/`, online as of September 2007

optical tracking. They produce two varieties of touch-enabled devices: panels that can be mounted over existing screens, and frames that can be mounted around large projected displays. Using one such frame, I implemented a small test Magic Lens interface that allowed the user to draw arbitrarily shaped 2D Magic Lenses with their finger. Although by no means a conclusive test of their success, the interface was a motivating example of what may be possible.

There are many applications for this type of interaction. One potential scenario is a large public display located at a travel bureau where interested travellers can approach a wall-sized screen that displays a large maps of the world. The user can activate a new Magic Lens through which they can filter information related to their personal travel plans, such as hotels in a Vancouver, an ice tour in Greenland and so on. In addition, the lens can be used to zoom in on a region (which is likely to be small on the world map), and offset a region (so multiple users can investigate the same region without competing for physical space). An illustration of how such a screen would be used is shown in Figure 9.2.

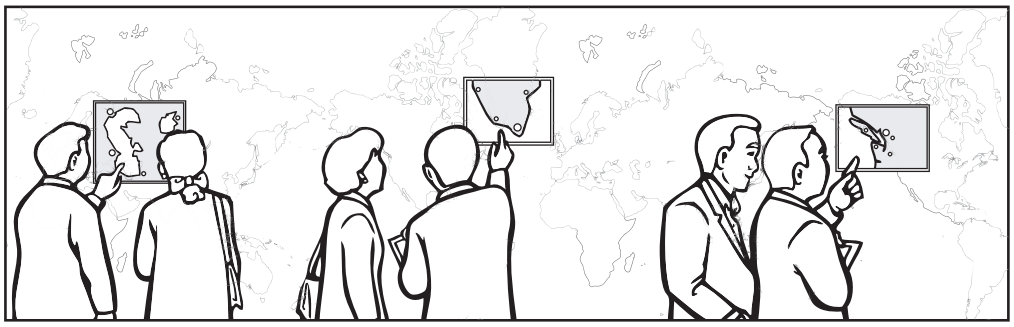


Figure 9.2: The TravelScreen

Another interesting type of display device that has been recently developed is the multi-layer display (MLD) described in Section 2.2.2. The MLD uses two physical display layers, separated by a small depth offset, two create a new type of display that literally present information in the foreground and background. Although there are presently some limitations to the technology, such as unintuitive colour blending between the two layers, there is

also the possibility that such displays will focus user attention to important information more effectively than a traditional single layer display.

Interaction techniques based on Magic Lenses may be well-suited to the MLD. The front display layer can be considered a transparent sheet on which Magic Lenses are placed, overlaying the information display space on the far layer. In this way, aspects of the ToolGlass interface described by Bier et al. (1993), in which tool palettes and Magic Lens filters sat in transparent layers over the workspace, would become a physical reality.

It is claimed that the depth cues provided by the MLD allow the user to disambiguate information more easily. Other than the feeling of depth provided by binocular vision, the MLD also provides motion-parallax (the way objects at different distances from the eye appear to move at slower speeds the further away they are). I believe it is this property of the MLD that would provide the most compelling effect on users. However, the MLD may not be the only way to provide such an experience.

Parallax can be simulated in software. In fact, a software approach allows many of the technical limitations of the MLD, such as confusing colour blending between the layers, to be avoided completely. One requirement, however, is that the head position of the user must be continuously known. There are several potential ways to achieve this, including face-tracking and inertial tracking. One relatively inexpensive solution is to use infra-red tracking devices like the TrackIR. The TrackIR device uses infra-red to track the position and orientation of the user's head. The user must wear a hat with a IR reflective strip of material sewn into it. The device and hat are intended for game players, who can experience an added sense of immersion in first-person games by being able to physically dodge, look around corners, and so on.

It would be interesting to investigate whether Magic Lenses, appearing to float a small distance above the workspace they are affecting, makes them any more or less effective as information visualisation tools. In such an evaluation, a standard display (no parallax) could by compared to real parallax provided by the MLD, and simulated parallax provided by custom software and the head tracker. These conditions are shown in Figure 9.3.

As shown in the survey of 3D Magic Lens systems in Section 2.3.2, 3D

(a) Standard Monitor (b) Multi-Layer Display (c) Software Parallax

Figure 9.3: Different Parallax Configurations.

Magic Lenses have been applied extensively in scientific visualisations, including flow and geographic visualisations. Some of the AR Magic Lens systems described in this thesis have been ported to the HIT Lab's VisionSpace system, which is also running Open Scene Graph based software. VisionSpace is a three-screen stereoscopic projection theatre with a wide-area infrared optical tracker, providing an immersive 3D visualisation environment. We have begun to explore how Magic Lenses can be exploited in this setting. Figure 9.4 shows an initial prototype of an X-Ray vision Magic Lens in action.

9.2 Future evaluations

There is still considerable work to be done on the evaluation of Magic Lenses. The 2D evaluation conducted in this thesis is a first step towards further understanding the interaction issues of Magic Lenses. The implementation challenges of Magic Lenses have been explored in more depth, and given the surplus of processing power available to the average user today, more efficient implementations seem less important than solid user interface guidelines.

The experiment in Chapter 4 investigated the Magic Lens as an information filtering approach. In this situation, the challenge to the user is increasing data density and the possibility of information overload.

Another experiment could be run to evaluate a different aspect of infor-

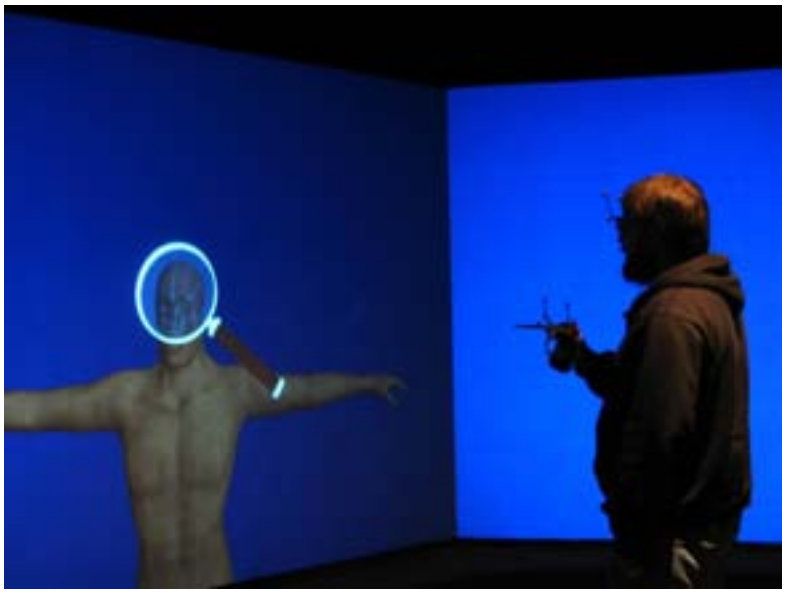


Figure 9.4: Magic Lenses in the VisionSpace visualisation system.

mation presentation: the lack of screen real-estate. The taxonomy presented in Section 2.2 identifies two general ways that the screen space can be partitioned: spatially and temporally. The most basic approaches in each case are panes, where the screen space is divided amongst multiple views, and tabs, where the screen space is devoted to one view at a time. A third approach is a Magic Lens that embeds a region of one view within another view.

Interestingly, augmented reality interfaces have the same challenges. In tangible AR, the analogies of panes and tabs are tiles and pages, and again, Magic Lens techniques can be used to compromise between these two approaches. These similarities are illustrated in Figure 9.5.

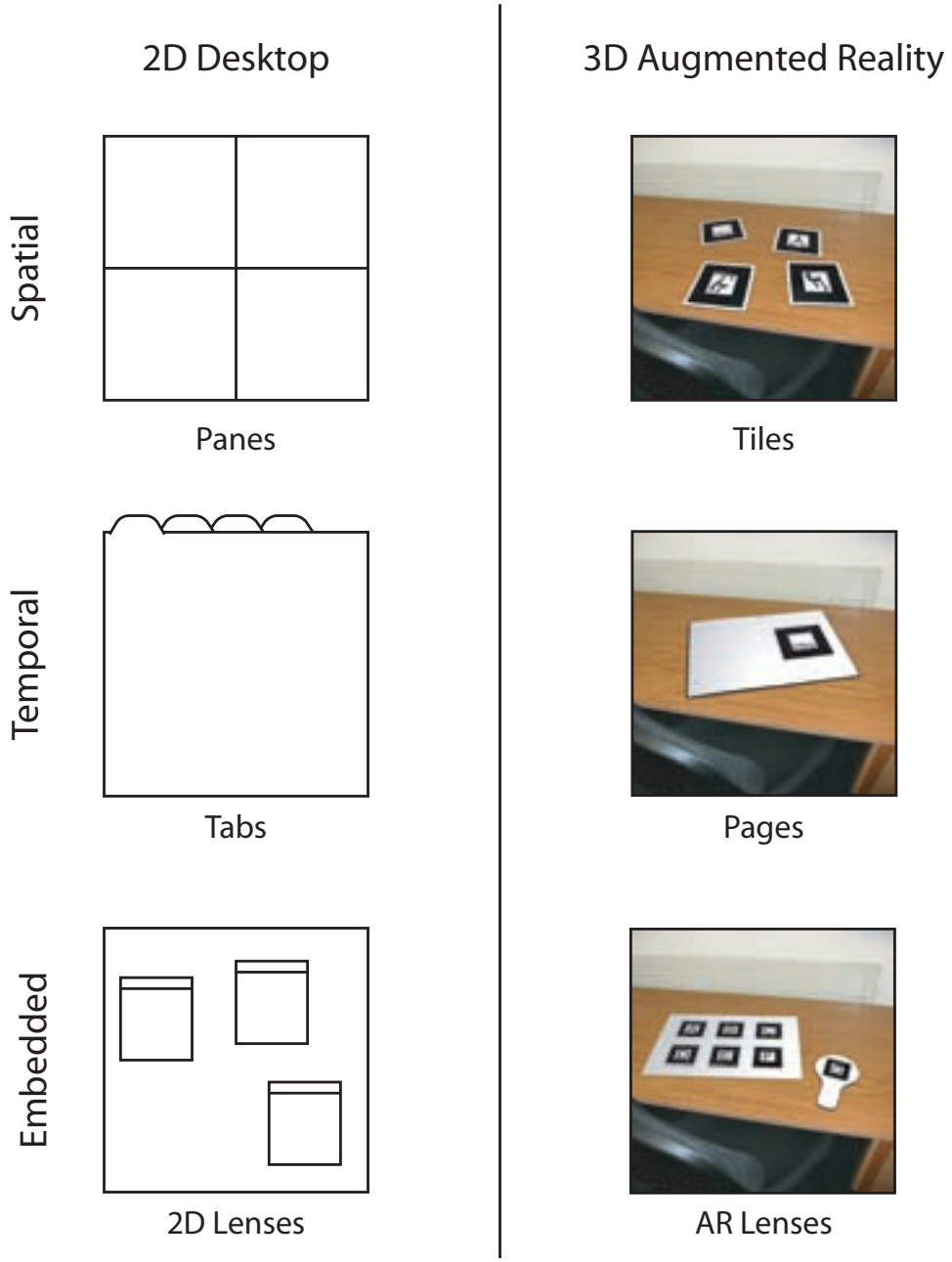


Figure 9.5: View separations in 2D and AR. 2D panes are analogous to tangible tiles in AR. The book metaphor common in AR is a temporal separation like tabs in 2D interfaces. Finally, lenses can provide a compromise between spatial and temporal view separations in both 2D and AR interfaces.

Chapter 10

Conclusion

Since their appearance in the early 1990s, there has been considerable effort put into implementing 2D Magic Lenses in many different domains, including graphic design, database querying, user interface development, and geographical information systems. However, there have been few, if any, formal evaluations of the effectiveness and efficiency of the technique. In Chapter 4, we conducted a formal evaluation of 2D Magic Lenses for a counting task in a mapping scenario. In this evaluation, the Magic Lens was compared to a technique that filtered the entire screen, and a technique that provided no filtering at all. Although the Magic Lens was the slowest technique, it was highly accurate and most preferred by participants. Participants felt confident and efficient with the Magic Lens, and claimed low frustration and low mental demand. In particular, they appreciated the Magic Lens' ability to provide an explicit area of focus within an otherwise cluttered workspace. A follow-up analysis showed that the less-than-optimal time performance of the Magic Lens technique was related to the high manipulation cost of resizing the lens. Enhancements to the design that could overcome this problem were suggested, such as using the scroll-wheel to resize the lens or using a spray-can metaphor to create lenses. Such proposed solutions will be the subject of future work.

As with all visualisation applications, tools and techniques are required to help the user manage information. Magic Lenses have been applied to this problem in various 2D interfaces. The next step is applying them within Augmented Reality where similar challenges of providing Focus and Context exist.

Augmented Reality user interfaces are blurring the boundary between the physical and virtual worlds. The view of the real world can be enhanced

through AR technology, and physical objects can be assigned digital meaning to support tangible interaction. Both of these features suit the Magic Lens technique, which can help the user manage visual complexity through the familiar magnifying glass metaphor. To this end, we explored existing work on 3D Magic Lenses and extended the implementations for AR. These implementations were developed in Open Scene Graph and we created a software library, osgART, to add the required video see-through AR capabilities. Using osgART, we created many AR Magic Lens prototypes, each exemplifying different potential applications, including data browsing, x-ray vision, entertainment and education. These projects demonstrated that AR Magic Lenses are intuitive to use, and suggest significant benefits when AR interfaces become more accessible and widespread.

Although the AR Magic Lens shows promise as a tool for visualisation, it is also important to consider how it can be used for other interface tasks. Once an object of interest is identified within the lens, the user is likely to wish to perform operations on it. In direct manipulation interfaces, object selection is a prerequisite for action, so the selection technique must be efficient. We implemented a selection technique for AR Magic Lenses and evaluated it against the traditional techniques of Direct Touch and Ray-Casting. The Magic Lens technique was faster than the other techniques, and required less head and hand motion to use. It was also the preferred technique, and participants found it the least physically and mentally demanding. These results, all statistically significant, are promising and encourage the further development of AR Magic Lenses.

In an exploration of what is possible with AR Magic Lenses, we developed a variation called Flexible Sheet Lenses. These are resizable visualisation surfaces suspended between two tracked handles. This configuration introduces new interaction techniques based on bimanual poses, which were recognised by our system using a neural network. The flexible lens' larger display area and dual handles make it more suitable for augmenting real objects than the previous implementation. In the circular AR Magic Lenses a tracking marker constantly obscured the real world behind it, whereas the flexible sheet hangs in the open space between the two handles. An informal user study gave positive feedback concerning the usability and enjoyability of the

system, although fatigue in the users' arms is an indication that the physical design of the handles requires further research. Also, the mappings between poses and interface actions need careful consideration. We extended the concept further to create flexible volume lenses, which could have applications in collaborative AR environments where it is difficult for multiple users to share a view-dependent technique like the flexible sheet.

As new technology trends emerge, such as multi-touch screens and large public displays, 2D Magic Lenses have the potential to enhance the usability and management of information within interfaces on these devices. In addition, as AR becomes more prevalent, Magic Lenses will address the challenge of interactive Focus and Context in Augmented Reality.

References

Ahlberg, C. and Shneiderman, B.: 1994a, Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays, *Proceedings of CHI 94 Conference on Human Factors in Computing Systems*, pp. 313 317.

Ahlberg, C. and Shneiderman, B.: 1994b, Visual Information Seeking Using the FilmFinder, *Proceedings of CHI 94 Conference on Human Factors in Computing Systems*, pp. 433 434.

Apple Computer: 1992, *Macintosh Human Interface Guidelines*, Addison-Wesley.

ARToolKit: 2001, ARToolKit website: http://www.hitl.washington.edu/artoolkit/.

Azuma, R., Hoff, B., Neely, H. and Sarfaty, R.: 1999, A Motion-Stabilized Outdoor Augmented Reality System, *Proceedings of IEEE VR 99 Conference on Virtual Reality*, pp. 252 259.

Azuma, R. T.: 1997, A Survey of Augmented Reality, *Presence: Teleoperators and Virtual Environments* **6**(4), 355 385.

Baldonado, M. Q. W., Woodruff, A. and Kuchinsky, A.: 2000, Guidelines for Using Multiple Views in Information Visualization, *Proceedings of AVI 00 Conference on Advanced Visual Interfaces*, pp. 110 119.

Balzer, M., Deussen, O. and Lewerentz, C.: 2005, Voronoi Treemaps for the Visualization of Software Metrics, *Proceedings of SoftVis 05 Symposium on Software Visualization*, pp. 165 172.

Baudisch, P., Good, N. and Stewart, P.: 2001, Focus Plus Context Screens: Combining Display Technology with Visualization Techniques, *Proceed-*

ings of UIST 01 Symposium on User Interface Software and Technology, pp. 31 40.

Baudisch, P. and Gutwin, C.: 2004, Multiblending: Displaying Overlapping Windows Simultaneously Without the Drawbacks of Alpha Blending, *Proceedings of CHI 2004 Conference on Human Factors in Computing Systems*, pp. 367 374.

Baudisch, P. and Rosenholtz, R.: 2003, Halo: A Technique for Visualizing Off-Screen Objects, *Proceedings of CHI 2003 Conference on Human Factors in Computing Systems*, pp. 481 488.

Bederson, B. B.: 2001, PhotoMesa: A Zoomable Image Browser using Quantum Treemaps and Bubblemaps, *Proceedings of UIST 01 Symposium on User Interface Software and Technology*, pp. 71 80.

Behzadan, A. H. and Kamat, V. R.: 2005, Visualization of Construction Graphics in Outdoor Augmented Reality, *Proceedings of WSC 05 Winter Simulation Conference*, pp. 1914 1920.

Bichlmeier, C., Sielhorst, T. and Navab, N.: 2006, The Tangible Virtual Mirror: New Visualization Paradigm for Navigated Surgery, *International Workshop on Augmented Reality Environments for Medical Imaging and Computer-Aided Surgery*, Copenhagen, Denmark.

Bier, E. A., Stone, M. C., Pier, K., Buxton, W. and DeRose, T. D.: 1993, Toolglass and MagicLenses: The See-Through Interface, *Proceedings of SIGGRAPH 93 Conference on Computer Graphics and Interactive Techniques*, pp. 73 80.

Bier, E., Stone, M. and Pier, K.: 1997, Enhanced Illustration Using Magic Lens Filters, *IEEE Computer Graphics and Applications* **17**(6), 62 70.

Billinghurst, M.: 2001, Usability Testing of Augmented/Mixed Reality Systems, *Technical report*, University of Washington.

Billinghurst, M., Kato, H. and Poupyrev, I.: 2001, The MagicBook - Moving Seamlessly between Reality and Virtuality, *IEEE Computer Graphics and Applications* **21**(3), 6 8.

Bimber, O., Encarnaçao, L. M. and Schmalstieg, D.: 2003, The Virtual Showcase as a New Platform for Augmented Reality Digital Storytelling, *Proceedings of EGVE 03 EuroGraphics Workshop on Virtual Environments 2003*, pp. 87 95.

Björk, S., Holmquist, L. E. and Redström, J.: 1999, A Framework for Focus+Context Visualization, *Proceedings of INFOVIS 99 Symposium on Information Visualization*, pp. 53 56.

Blanch, R., Guiard, Y. and Beaudouin-Lafon, M.: 2004, Semantic Pointing: Improving Target Acquisition with Control-Display Ratio Adaptation, *Proceedings of CHI 2004 Conference on Human Factors in Computing Systems*, pp. 519 526.

Boeck, J. D., Weyer, T. D., Raymaekers, C. and Coninx, K.: 2006, Using the Non-Dominant Hand for Selection in 3D, *Proceedings of 3DUI 06 Symposium on 3D User Interfaces*, pp. 53 58.

Bowman, D. A. and Hodges, L. F.: 1999, Formalizing the Design, Evaluation and Application of Interaction Techniques for Immersive Virtual Environments, *Visual Languages and Computing* **10**(1), 37 53.

Bowman, D. A., Kruijff, E., Joseph J. LaViola, J. and Poupyrev, I.: 2004, *3D User Interfaces Theory and Practice*, Addison-Wesley.

Bowman, D., Johnson, D. and Hodges, L.: 2001, Testbed Evaluation of Virtual Environment Interaction Techniques, *Presence: Teleoperators and Virtual Environments* **10**(1), 75 95.

Brooke, J.: 1996, *SUS: A Quick and Dirty Usability Scale.*, Taylor and Francis, London, England, pp. 189 194.

Brown, L. D., Hua, H. and Gao, C.: 2003, A Widget Framework for Augmented Interaction in SCAPE, *Proceedings of UIST 03 Symposium on User Interface Software and Technology*, pp. 1 10.

Brown, L. and Hua, H.: 2006, Magic Lenses for Augmented Virtual Environments, *IEEE Computer Graphics and Applications* **26**(4), 64 73.

Buchmann, V., Violich, S., Billinghurst, M. and Cockburn, A.: 2004, FingARtips: Gesture Based Direct Manipulation in Augmented Reality, *Proceedings of GRAPHITE 04 Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia*, pp. 212 221.

Burbeck, S.: 1992, Applications Programming in Smalltalk-80^{TM}: How to use Model-View-Controller (MVC), Published online, http://st-www.cs.uiuc.edu/users/smarch/st-docs/mvc.html.

Buschmann, F., Meunier, R., Rohnert, H., Sommerlan, P. and Stal, M.: 1996, *Pattern-Oriented Software Architecture*, John Wiley and Sons.

Card, S., MacKinlay, J. and Shneiderman, B.: 1999, *Readings in Information Visualisation: Using Vision to Think*, Morgan Kaufmann.

Caudell, T. P. and Mizell, D. W.: 1992, Augmented Reality: An Application of Heads-Up Display Technology to Manual Manufacturing Processes, *Proceedings of IEEE Hawaii International Conference on Systems Sciences*, pp. 659 669.

Cavallaro, R.: 1997, The FoxTrax Hockey Puck Tracking System, *IEEE Computer Graphics and Applications* **17**(2), 6 12.

Cignoni, P., Montani, C. and Scopigno, R.: 1994, MagicSphere: an Insight Tool for 3D Data Visualization, *Computer Graphics Forum* **13**(3), 317 328.

Cockburn, A., Gutwin, C. and Alexander, J.: 2006, Faster Document Navigation with Space-Filling Thumbnails, *Proceedings of CHI 2006 Conference on Human Factors in Computing Systems*, pp. 1 10.

Cockburn, A., Looser, J. and Savage, J.: 2003, Around the World in Seconds with Speed-Dependent Automatic Zooming, *Proceedings of UIST 03 Symposium on User Interface Software and Technology*, pp. 35 36.

Coelho, E. M., MacIntyre, B. and Julier, S.: 2004, OSGAR: A Scene Graph with Uncertain Transformations, *Proceedings of ISMAR 04 International Symposium on Mixed and Augmented Reality 2004*, pp. 6 15.

Cox, D. A., Chugh, J. S., Gutwin, C. and Greenberg, S.: 1998, The Usability of Transparent Overview Layers, *Proceedings of CHI 98 Conference on Human Factors in Computing Systems*, pp. 301 302.

de Bruijn, O. and Spence, R.: 2000, Rapid Serial Visual Presentation: A Space-Time Trade-Off in Information Presentation, *Proceedings of AVI 00 Conference on Advanced Visual Interfaces*, pp. 189 192.

de Bruijn, O. and Spence, R.: 2002, Patterns of Eye Gaze during Rapid Serial Visual Presentation, *Proceedings of AVI 02 Conference on Advanced Visual Interfaces*, pp. 209 217.

Dias, J., Barata, N., Santos, P., Correia, A., Nande, P. and Bastos, R.: 2003, In Your Hand Computing: Tangible Interfaces for Mixed Reality, *Proceedings of the Augmented Reality Toolkit Workshop 2003*, pp. 29 31.

DiVerdi, S., Nurmi, D. and Höllerer, T.: 2003, ARWin - A Desktop Augmented Reality Window Manager, *Proceedings of ISMAR 03 International Symposium on Mixed and Augmented Reality 2004*, p. 298.

Döllner, J., Baumman, K. and Hinrichs, K.: 2000, Texturing Techniques for Terrain Visualization, *Proceedings of IEEE Visualization 00 Conference on Visualization*, pp. 227 234.

Drascic, D. and Milgram, P.: 1996, Perceptual Issues in Augmented Reality, *Proceeding of SPIE: Stereoscopic Displays and Virtual Reality Systems 1996*, Vol. 2653, pp. 123 134.

Dreyfuss, H.: 1967, *The Measure of Man. Human Factors in Design*, Whitney Library of Design, New York, New York, United States.

Ellis, G., Bertini, E. and Dix, A.: 2005, The Sampling Lens: Making Sense of Saturated Visualisations, *Proceedings of CHI 2005 Conference on Human Factors in Computing Systems*, pp. 1351 1354.

Elvins, T. T., Nadeau, D. R., Schul, R. and Kirsh, D.: 1998, Worldlets: 3D Thumbnails for 3D Browsing, *Proceedings of CHI 98 Conference on Human Factors in Computing Systems*, pp. 163 170.

Engdahl, B., Köksal, M. and Marsden, G.: 2005, Using Treemaps to Visualize Threaded Discussion Forums on PDAs, *Proceedings of CHI 2005 Conference on Human Factors in Computing Systems*, pp. 1355 1358.

Fishkin, K. and Stone, M. C.: 1995, Enhanced Dynamic Queries via Movable Filters, *Proceedings of CHI 95 Conference on Human Factors in Computing Systems*, pp. 415 420.

Fitts, P. M.: 1954, The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement, *Journal of Experimental Psychology* **47**(6), 381 391.

Fitzmaurice, G. W. and Buxton, W.: 1997, An Empirical Evaluation of Graspable User Interfaces: Towards Specialized, Space-Multiplexed Input, *Proceedings of CHI 97 Conference on Human Factors in Computing Systems*, pp. 43 50.

Fjeld, M., Juchli, P. and Voegtli, B. M.: 2003, Chemistry Education: A Tangible Interaction Approach, *Proceedings of INTERACT 03 Conference on Human-Computer Interaction*, pp. 287 294.

Forbes, K.: 2007, cWiiMote website: http://simulatedcomicproduct.com/2006/12/cwiimote-02.php.

Forsberg, A., Herndon, K. and Zeleznik, R.: 1996, Aperture Based Selection for Immersive Virtual Environments, *Proceedings of UIST 96 Symposium on User Interface Software and Technology*, pp. 95 96.

Forsythe, T.: 2001, *Game Programming Gems 2*, Charles River Media, Massachusetts, United States, chapter Imposters: Adding Clutter, pp. 488 496.

Fox, D.: 1998, Composing Magic Lenses, *Proceedings of CHI 98 Conference on Human Factors in Computing Systems*, pp. 519 525.

Friedrich, W.: 2002, ARVIKA Augmented Reality for Development, Production and Service, *Proceedings of ISMAR 02 International Symposium on Mixed and Augmented Reality 2004*, p. 3.

Fröhlich, B., Barrass, S., Zehner, B., Plate, J. and Göbel, M.: 1999, Exploring Geo-Scientific Data in Virtual Environments, *Proceedings of IEEE Visualization 99 Conference on Visualization*, pp. 169 173.

Fuhrmann, A. L. and Gröller, E.: 1998, Real-Time Techniques for 3D Flow Visualization, *Proceedings of IEEE Visualization 98 Conference on Visualization*, pp. 305 312.

Furnas, G. W.: 1986, Generalized Fisheye Views, *Proceedings of CHI 86 Conference on Human Factors in Computing Systems*, pp. 16 23.

Furnas, G. W.: 2006, A Fisheye Follow-Up: Further Reflections on Focus+Context, *Proceedings of CHI 2006 Conference on Human Factors in Computing Systems*, pp. 999 1008.

Furness, T.: 1986, The Super Cockpit and Human Factors Challenges, *Proceedings of Human Factors Society 30th Annual Meeting*, pp. 48 52.

Gaylin, K. B.: 1986, How Are Windows Used? Some Notes on Creating an Empirically-Based Windowing Benchmark Task, *Proceedings of CHI 86 Conference on Human Factors in Computing Systems*, pp. 96 100.

Gibson, J. J.: 1979, *The Ecological Approach To Visual Perception*, Houghton Mifflin.

Gillet, A., Sanner, M., Stoffler, D. and Olson, A.: 2005, Tangible Augmented Interfaces for Structural Molecular Biology, *IEEE Computer Graphics and Applications* **25**(2), 13 17.

GmbH, A. R. T.: 2007, ART website: http://www.ar-tracking.de.

Grasset, R., Looser, J. and Billinghurst, M.: 2005, OSGARToolKit: Tangible + Transitional 3D Collaborative Mixed Reality Framework, *Proceedings of ICAT 05 Conference on Artificial Reality and Telexistence*, pp. 257 258.

Greenberg, S., Gutwin, C. and Cockburn, A.: 1996, Using Distortion-Oriented Displays to Support Workspace Awareness, *Proceedings of People and Computers XI: British Computer Society Conference on Human Computer Interaction*, pp. 299 314.

Grosjean, J. and Coquillart, S.: 1999, The Magic Mirror: A Metaphor for Assisting the Exploration of Virtual Worlds, *Proceedings of the 1999 Spring Conference on Computer Graphics*, pp. 125 129.

Grossman, T. and Balakrishnan, R.: 2006, The Design and Evaluation of Selection Techniques for 3D Volumetric Displays, *Proceedings of UIST 06 Symposium on User Interface Software and Technology*, pp. 3 12.

Guiard, Y.: 1987, Asymmetric Division of Labor in Human Skilled Bimanual Action: The Kinematic Chain as a Model, *Journal of Motor Behavior* **19**(4), 486 517.

Haller, M., Hartmann, W., Luckeneder, T. and Zauner, J.: 2002, Combining ARToolKit with Scene Graph Libraries, *Proceedings of the Augmented Reality Toolkit Workshop 2002*, p. 2.

Haniff, D. J. and Baber, C.: 2003, User Evaluation of Augmented Reality Systems, *Proceedings of IV 03 Conference on Computer Visualization and Graphics Applications*.

Hart, S. and Staveland, L.: 1988, Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research, *in* P. Hancock and N. Meshkati (eds), *Human Mental Workload*, Elsevier Science, pp. 139 183.

Hedley, N. R., Billinghurst, M., Postner, L., May, R. and Kato, H.: 2002, Explorations in the use of Augmented Reality for Geographic Visualization, *Presence: Teleoperators and Virtual Environments* **11**(2), 119 133.

Heidmann, T.: 1991, Real Shadows, Real Time, *Iris Universe* **18**, 23 31.

Henrysson, A., Ollila, M. and Billinghurst, M.: 2005, Mobile Phone Based AR Scene Assembly, *Proceedings of MUM 05 Conference on Mobile and Ubiquitous Multimedia*, pp. 95 102.

Hinckley, K., Pausch, R., Goble, J. C. and Kassell, N. F.: 1994, Passive Real-World Interface Props for Neurosurgical Visualization, *Proceedings of SIGGRAPH 94 Conference on Computer Graphics and Interactive Techniques*, pp. 452 458.

Holman, D., Vertegaal, R., Altosaar, M. and Johns, N. T. D.: 2005, Paper Windows: Interaction Techniques for Digital Paper, *Proceedings of CHI 2005 Conference on Human Factors in Computing Systems*, pp. 591 599.

Hua, H., Gao, C. and Brown, L. D.: 2003, A New Collaborative Infrastructure: SCAPE, *Proceedings of IEEE VR 03 Conference on Virtual Reality*, pp. 171 179.

Hudson, S. E., Rodenstein, R. and Smith, I.: 1997, Debugging Lenses: A New Class of Transparent Tools for User Interface Debugging, *Proceed-*

ings of UIST 97 Symposium on User Interface Software and Technology, pp. 179 187.

Igarashi, T. and Hinckley, K.: 2000, Speed-Dependent Automatic Zooming for Browsing Large Documents, *Proceedings of UIST 00 Symposium on User Interface Software and Technology*, pp. 139 148.

Ishak, E. W. and Feiner, S. K.: 2004, Interacting with Hidden Content using Content-Aware Free-Space Transparency, *Proceedings of UIST 04 Symposium on User Interface Software and Technology*, pp. 189 192.

Ishii, H. and Ullmer, B.: 1997, Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms, *Proceedings of CHI 97 Conference on Human Factors in Computing Systems*, pp. 234 241.

Johnson, B. and Shneiderman, B.: 1991, Tree-Maps: A Space-Filling Approach to the Visualization of Hierarchical Information Structures, *Proceedings of IEEE Visualization 91 Conference on Visualization*, pp. 284 291.

Johnson, J.: 2000, *GUI Bloopers: Don ts and Do s for Software Developers and Web Designers*, Morgan Kaufmann.

Kalghatgi, N., Burgman, A., Darling, E., Newbern, C., Recktenwald, K., Chin, S. and Kong, H.: 2006, Geospatial Intelligence Analysis via Semantic Lensing, *Proceedings of CHI 2006 Conference on Human Factors in Computing Systems*, pp. 935 940.

Kato, H. and Billinghurst, M.: 1999, Marker Tracking and HMD Calibration for a Video-Based Augmented Reality Conferencing System, *Proceedings of IWAR 99 International Workshop on Augmented Reality*, p. 85.

Kato, H., Billinghurst, M., Poupyrev, I. and Tetsutani, N.: 2001, Tangible Augmented Reality for Human Computer Interaction, *Proceedings of NICOGRAPH 01*, pp. 39 44.

Kato, H., Tachibana, K., Tanabe, M., Nakajima, T. and Fukuda, Y.: 2003, MagicCup: A Tangible Interface for Virtual Object Manipulation in Table-Top Augmented Reality, *Proceedings of the Augmented Reality Toolkit Workshop 2003*, pp. 75 76.

Kilgard, M. J.: 1999, Improving Shadows and Reflections via the Stencil Buffer, *Advanced Game Development course notes, Game Developers Conference*, pp. 204 253.

King, G. R., Piekarski, W. and Thomas, B. H.: 2005, ARVino: Outdoor Augmented Reality Visualisation of Viticulture GIS Data, *Proceedings of ISMAR 05 International Symposium on Mixed and Augmented Reality 2005*, pp. 52 55.

Kirsch, F. and Döllner, J.: 2004, Rendering Techniques for Hardware-Accelerated Image-Based CSG, *Journal of WSCG* **12**(2), 221 228.

Kitamura, Y., Itoh, Y. and Kishino, F.: 2001, Real-Time 3D Interaction with ActiveCube, *Proceedings of CHI 2001 Conference on Human Factors in Computing Systems*, pp. 355 356.

Kiyokawa, K., Takemura, H. and Yokoya, N.: 1999, A Collaboration Support Technique by Integrating a Shared Virtual Reality and a Shared Augmented Reality, *Proceedings of IEEE SMC 99 Conference on Systems, Man, and Cybernetics*, Vol. 6, pp. 48 53.

Kosara, R., Miksch, S. and Hauser, H.: 2001, Semantic Depth of Field, *Proceedings of INFOVIS 01 Symposium on Information Visualization*, pp. 22 23.

Kosara, R., Miksch, S. and Hauser, H.: 2002, Focus+Context Taken Literally, *IEEE Computer Graphics and Applications* **22**(1), 22 29.

Lamping, J., Rao, R. and Pirolli, P.: 1995, A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies, *Pro-*

ceedings of CHI 95 Conference on Human Factors in Computing Systems, pp. 401 408.

Lee, G. A., Billinghurst, M. and Kim, G. J.: 2004, Occlusion Based Interaction Methods for Tangible Augmented Reality Environments, *Proceedings of VRCAI 04 Conference on the Virtual Reality Continuum and its Applications in Industry*, pp. 419 426.

Leung, Y. K. and Apperley, M. D.: 1994, A Review and Taxonomy of Distortion-Oriented Presentation Techniques, *ACM Trans. Comput.-Hum. Interact.* **1**(2), 126 160.

Liang, J. and Green, M.: 1994, JDCAD: A Highly Interactive 3D Modeling System, *Computers and Graphics* **18**(4), 499 506.

Looser, J., Billinghurst, M. and Cockburn, A.: 2004, Through the Looking Glass: The Use of Lenses as an Interface Tool for Augmented Reality Interfaces, *Proceedings of GRAPHITE 04 Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia*, pp. 204 211.

Looser, J., Grasset, R., Seichter, H. and Billinghurst, M.: 2006, OSGART - A Pragmatic Approach to MR, *In Industrial Workshop at ISMAR 2006*.

MacKenzie, I. S.: 1995, *Movement Time Prediction in Human-Computer Interfaces*, Morgan Kaufmann Publishers Inc., San Francisco, California, United States, pp. 483 492.

Mackinlay, J. D., Robertson, G. G. and Card, S. K.: 1991, The Perspective Wall: Detail and Context Smoothly Integrated, *Proceedings of CHI 91 Conference on Human Factors in Computing Systems*, pp. 173 176.

Mason, A. H., Walji, M. A., Lee, E. J. and MacKenzie, C. L.: 2001, Reaching Movements to Augmented and Graphic Objects in Virtual Environments, *Proceedings of CHI 2001 Conference on Human Factors in Computing Systems*, pp. 426 433.

Masoodian, M., McKoy, S., Rogers, B. and Ware, D.: 2004, DeepDocument: Use of a Multi-Layered Display to Provide Context Awareness in Text Editing, *Proceedings of AVI 04 Conference on Advanced Visual Interfaces*, pp. 235 239.

Microsoft Corporation: 1999, *Microsoft Windows User Experience (Microsoft Professional Edition)*, Microsoft Press, Redmond, Washington, United States.

Milgram, P. and Kishino, F.: 1994, Augmented Reality: A Class of Displays on the Reality-Virtuality Continuum, *Proceedings of SPIE: Telemanipulator and Telepresence Technologies 1994*, pp. 42 48.

Moore, A. and Regenbrecht, H.: 2005, The Tangible Augmented Street Map, *Proceedings of ICAT 05 Conference on Artificial Reality and Telexistence*, pp. 249 250.

Napari, H.: 1999, *Magic Lens User Interface in Virtual Reality*, Master's thesis, Helsinki University of Technology.

Navab, N., Cubillo, E., Bascle, B., Lockau, J., Kamsties, K.-D. and Neuberger, M.: 1999, CyliCon: A Software Platform for the Creation and Update of Virtual Factories, *Proceedings of ETFA 99 Conference on Emerging Technologies and Factory Automation*, pp. 459 463.

Nikishkov, G. and Tsuchimoto, T.: 2007, Using augmented reality for real-time visualization of tactile health examination, *Proceedings of GRAPP 07 Conference on Computer Graphics Theory and Applications*, pp. 91 97.

Nilsen, T.: 2005, Tankwar: AR games at GenCon Indy 2005, *Proceedings of ICAT 05 Conference on Artificial Reality and Telexistence*, pp. 243 244.

Nilsen, T.: 2006, *Guidelines for the Design of Augmented Reality Strategy Games*, Master's thesis, University of Canterbury.

Nilsen, T., Linton, S. and Looser, J.: 2004, Motivations for AR Gaming, *Proceedings of Fuse 04 New Zealand Game Developers Conference*, pp. 86 93.

Nissen, S.: 2003, Implementation of a Fast Artificial Neural Network Library (FANN), *Technical report*, Department of Computer Science, University of Copenhagen.

Osfield, R.: 2007, Open Scene Graph website: http://www.openscenegraph.org.

Pang, A., Wittenbrink, C. M. and Goodman, T.: 1995, CSpray: A Collaborative Scientific Visualization Application, *Proceedings of SPIE: Multimedia Computing and Networking 1995*, pp. 317 326.

Perlin, K. and Fox, D.: 1993, Pad: An Alternative Approach to the Computer Interface, *Proceedings of SIGGRAPH 93 Conference on Computer Graphics and Interactive Techniques*, pp. 57 64.

Piekarski, W.: 2004, *Interactive 3D Modelling in Outdoor Augmented Reality Worlds*, PhD thesis, School of Computer and Information Science, University of South Australia.

Piekarski, W. and Thomas, B. H.: 2002, The Tinmith System: Demonstrating New Techniques for Mobile Augmented Reality Modelling, *Proceedings of AUIC 02 Australasian Conference on User Interfaces*, pp. 61 70.

Pierce, J. S., Forsberg, A. S., Conway, M. J., Hong, S., Zeleznik, R. C. and Mine, M. R.: 1997, Image Plane Interaction Techniques in 3D Immersive Environments, *Proceedings of SI3D 97 Symposium on Interactive 3D Graphics*, pp. 39 44.

Platt, J. R. H. and Biesty, S.: 1992, *Incredible Cross-Sections*, Dorling Kindersley, London, England.

Poupyrev, I., Billinghurst, M., Weghorst, S. and Ichikawa, T.: 1996, The Go-Go Interaction Technique: Non-Linear Mapping for Direct Manipulation in VR, *Proceedings of UIST 96 Symposium on User Interface Software and Technology*, pp. 79 80.

Poupyrev, I., Ichikawa, T., Weghorst, S. and Billinghurst, M.: 1998, Egocentric Object Manipulation in Virtual Environments: Empirical Evaluation of Interaction Techniques, *Computer Graphics Forum* **17**(3), 41 52.

Poupyrev, I., Tan, D. S., Billinghurst, M., Kato, H., Regenbrecht, H. and Tetsutani, N.: 2001, Tiles: A Mixed Reality Authoring Interface, *Proceedings of INTERACT 01 Conference on Human-Computer Interaction*, pp. 334 341.

Ramos, G., Cockburn, A., Balakrishnan, R. and Beaudouin-Lafon, M.: 2007, Pointing Lenses: Facilitating Stylus Input through Visual-and Motor-space Magnification, *Proceedings of CHI 2007 Conference on Human Factors in Computing Systems*, pp. 757 766.

Rao, R. and Card, S. K.: 1994, The Table Lens: Merging Graphical and Symbolic Representations in an Interactive Focus+Context Visualization for Tabular Information, *Proceedings of CHI 94 Conference on Human Factors in Computing Systems*, pp. 318 322.

Rekimoto, J.: 1995, The Magnifying Glass Approach to Augmented Reality Systems, *Proceedings of ICAT 95 Conference on Artificial Reality and Telexistence*, pp. 123 132.

Robertson, G. G. and Mackinlay, J. D.: 1993, The Document Lens, *Proceedings of UIST 93 Symposium on User Interface Software and Technology*, pp. 101 108.

Robertson, G. G., Mackinlay, J. D. and Card, S. K.: 1991, Cone Trees: Animated 3D Visualizations of Hierarchical Information, *Proceedings of CHI 91 Conference on Human Factors in Computing Systems*, pp. 189 194.

Ropinski, T. and Hinrichs, K.: 2004, Real-Time Rendering of 3D Magic Lenses Having Arbitrary Convex Shapes, *Journal of the International Winter School of Computer Graphics (WSCG04)* **12**(1-3), 379 386.

Ropinski, T., Hinrichs, K. and Steinicke, F.: 2005, A Solution for the Focus and Context Problem in Interactive Geovisualization Applications, *Proceedings of DMGIS 05 Workshop on Dynamic and Multi-dimensional GIS*, pp. 144 149.

Ryall, K., Li, Q. and Esenther, A.: 2005, Temporal Magic Lens: Combined Spatial and Temporal Query and Presentation, *Proceedings of INTERACT 05 Conference on Human-Computer Interaction*, pp. 809 822.

Sarkar, M. and Brown, M. H.: 1992, Graphical Fisheye Views of Graphs, *Proceedings of CHI 92 Conference on Human Factors in Computing Systems*, pp. 83 91.

Sarkar, M., Snibbe, S. S., Tversky, O. J. and Reiss, S. P.: 1993, Stretching the Rubber Sheet: A Metaphor for Viewing Large Layouts on Small Screens, *Proceedings of UIST 93 Symposium on User Interface Software and Technology*, pp. 81 91.

Schaufler, G. and Schmalstieg, D.: 1999, Sewing Worlds Together With SEAMS: A Mechanism To Construct Large-Scale Virtual Environments, *Presence: Teleoperators and Virtual Environments* **8**(4), 449 461.

Schmalstieg, D., Encarnaçao, L. M. and Szalavári, Z.: 1999, Using Transparent Props for Interaction With The Virtual Table, *Proceedings of SI3D 99 Symposium on Interactive 3D Graphics*, pp. 147 153.

Schwesig, C., Poupyrev, I. and Mori, E.: 2004, Gummi: A Bendable Computer, *Proceedings of CHI 2004 Conference on Human Factors in Computing Systems*, pp. 263 270.

Segal, M., Korobkin, C., van Widenfelt, R., Foran, J. and Haeberli, P.: 1992, Fast Shadows and Lighting Effects using Texture Mapping, *Proceedings*

of SIGGRAPH 92 Conference on Computer Graphics and Interactive Techniques, pp. 249 252.

Shelton, B. and Hedley, N.: 2002, Using Augmented Reality for Teaching Earth-Sun Relationships to Undergraduate Geography Students, *The First IEEE International Augmented Reality Toolkit Workshop.*

Shneiderman, B.: 1987, *Direct Manipulation: A Step Beyond Programming Languages*, Morgan Kaufmann Publishers Inc., San Francisco, California, United States, pp. 461 467.

Shneiderman, B.: 1992, Tree Visualization with Tree-Maps: A 2-D Space-Filling Approach, *ACM Transactions on Graphics* **11**(1), 92 99.

Soler, L., Nicolau, S., Schmid, J., Koehl, C., Marescaux, J., Pennec, X. and Ayache, N.: 2004, Virtual Reality and Augmented Reality in Digestive Surgery, *Proceedings of ISMAR 04 International Symposium on Mixed and Augmented Reality 2004*, pp. 278 279.

Steed, A. and Parker, C.: 2004, 3D Selection Strategies for Head Tracked and Non-Head Tracked Operation of Spatially Immersive Displays, *Proceedings of IPT 04 Workshop on Immersive Projection Technology.*

Stoakley, R., Conway, M. J. and Pausch, R.: 1995, Virtual Reality on a WIM: Interactive Worlds in Miniature, *Proceedings of CHI 95 Conference on Human Factors in Computing Systems*, pp. 265 272.

Stoev, S. L., Schmalstieg, D. and Straßer, W.: 2002, The Through-The-Lens Metaphor: Taxonomy and Application, *Proceedings of IEEE VR 02 Conference on Virtual Reality*, pp. 285 286.

Stone, M., Fishkin, K. and Bier, E.: 1994, The Movable Filter as a User Interface Tool, *Proceedings of CHI 94 Conference on Human Factors in Computing Systems*, pp. 306 312.

Sutherland, I.: 1968, A Head-Mounted Three Dimensional Display, *Proceedings of the Fall Joint Computer Conference 1968*, Thompson Books, Washington, DC, pp. 757 764.

Tan, D. S., Robertson, G. G. and Czerwinski, M.: 2000, Exploring 3D Navigation: Combining Speed-Coupled Flying with Orbiting, *Proceedings of CHI 2001 Conference on Human Factors in Computing Systems*, pp. 418 425.

Tripathi, A.: 2000, *Augmented Reality: An Application for Architecture*, Master's thesis, University of Southern California.

Venkatasubramanian, S.: 2003, The Graphics Card as a Streaming Computer, *In SIGMOD Workshop on Management and Processing of Massive Data*.

Viega, J., Conway, M. J., Williams, G. and Pausch, R.: 1996, 3D Magic Lenses, *Proceedings of UIST 96 Symposium on User Interface Software and Technology*, pp. 51 58.

Vlahakis, V., Karigiannis, J., Tsotros, M., Gounaris, M., Almeida, L., Stricker, D., Gleue, T., Christou, I. T., Carlucci, R. and Ioannidis, N.: 2001, Archeoguide: First Results of an Augmented Reality, Mobile Computing System in Cultural Heritage Sites, *Proceedings of VAST 01 Conference on Virtual Reality, Archeology and Cultural Heritage*, pp. 131 140.

Ware, C.: 2004, *Information Visualization: Perception for Design*, 2 edn, Morgan Kaufmann.

Ware, C. and Lewis, M.: 1995, The DragMag Image Magnifier, *Proceedings of CHI 95 Conference on Human Factors in Computing Systems*, pp. 407 408.

Warmerdam, F.: 2007, Shapefile C Library V1.2 website: http://shapelib.maptools.org/.

Williamson, C. and Shneiderman, B.: 1992, The Dynamic HomeFinder: Evaluating Dynamic Queries in a Real-Estate Information Exploration System, *Proceedings of SIGIR 92 Conference on Research and Development in Information Retrieval*, pp. 338 346.

Wloka, M. M. and Greenfield, E.: 1995, The Virtual Tricorder: A Uniform Interface for Virtual Reality, *Proceedings of UIST 95 Symposium on User Interface Software and Technology*, pp. 39 40.

Wong, B. L. W., Joyekurun, R., Mansour, H., Amaldi, P., Nees, A. and Villanueva, R.: 2005, Depth, Layering and Transparency: Developing Design Techniques, *Proceedings of OZCHI 05 Australian Conference on Human Computer Interaction*, pp. 1 10.

Wong, N., Carpendale, S. and Greenberg, S.: 2003, EdgeLens: An Interactive Method for Managing Edge Congestion in Graphs, *Proceedings of INFOVIS 03 Symposium on Information Visualization*, pp. 51 58.

Wynn, C.: 2002, OpenGL Render-To-Texture, Presentation at Game Developers Conference, 2002.

Zellweger, P. T., Mackinlay, J. D., Good, L., Stefik, M. and Baudisch, P.: 2003, City Lights: Contextual Views in Minimal Space, *Proceedings of CHI 2003 Conference on Human Factors in Computing Systems*, pp. 838 839.

Zhou, Z., Cheok, A. D., Pan, J. and Li, Y.: 2004, Magic Story Cube: An Interactive Tangible Interface for Storytelling, *Proceedings of ACE 04 Conference on Advances in Computer Entertainment Technology*, pp. 364 365.

Appendix A

2D Experiment Questionnaire

Pre-Experiment Questionnaire

Subject: _____

Sex: M / F *(Please circle)*

Age: _____ years

Are you ☐ Left Handed or ☐ Right Handed?

Approximately how long do you normally spend in front of a computer screen per day? _____ hours

In general, do you mainly use the mouse or keyboard shortcuts during your work with computers?

☐ Mainly mouse ☐ Mainly keyboard ☐ Both the same

Do you play computer games? If so, how many hours per week do you devote to game playing?

☐ Don't play games ☐ Play games for _____ hours a week

Post-Condition Questionnaire for Low Density

Subject: _____

		Static	Global	Magic Lens
Q1	I found the interface easy to understand	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q2	I found the tasks easy to complete	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q3	I feel that I performed quickly	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q4	I feel confident about my accuracy	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q5	If I had to carry out this sort of work on a regular basis, this is an interface I would appreciate using	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q6	I found this condition physically demanding	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q7	I found this condition mentally demanding	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q8	I found this condition frustrating	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Please add any comments about the condition.				
When you have finished all three conditions.... Rank the conditions 1-3: (1 = most preferred, 3 = least preferred)				

Post-Condition Questionnaire for High Density

Subject: _____

		Static	Global	Magic Lens
Q1	I found the interface easy to understand	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q2	I found the tasks easy to complete	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q3	I feel that I performed quickly	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q4	I feel confident about my accuracy	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q5	If I had to carry out this sort of work on a regular basis, this is an interface I would appreciate using	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q6	I found this condition physically demanding	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q7	I found this condition mentally demanding	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q8	I found this condition frustrating	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Please add any comments about the condition.				
When you have finished all three conditions.... Rank the conditions 1-3: (1 = most preferred, 3 = least preferred)				

Post-Experiment Questionnaire

Subject: _____

Overall Preference

Please provide an overall rank for the interfaces you used. Write a number (1-3) next to each, with 1 being most preferred and 3 being least preferred.

_____ Single static view

_____ Local filtering (Magic Lens)

_____ Global filtering (Checkboxes)

Try to list three things you liked about the interface you marked with a 1:

Try to list three things you disliked about the interface you marked with a 3:

Have you ever encountered a Magic Lens interface before?

☐ No, this was the first time.

☐ Yes, in: *Tick any that apply and provide details if you can.*

- ☐ A research application
- ☐ A commercial application
- ☐ an academic paper, textbook or website
- ☐ fiction (movies, TV, books...)
- ☐ something else?

Details:

Please add any other comments or feedback here:

Appendix B

AR Experiment Questionnaire

Pre-Experiment Questionnaire

Subject: _____

Sex: M / F *(Please circle)*

Age: _____ years

Are you ☐ Left Handed or ☐ Right Handed?

How familiar are you with Augmented Reality (AR) interfaces?

☐ ☐ ☐ ☐ ☐ ☐ ☐

Unfamiliar Familiar

Post-Condition Questionnaire for Low Density Conditions

Subject: _____

	With respect to this low-density condition...	**Direct Touch**	**Ray Casting**	**Magic Lens**
Q1	I found the selection technique easy to understand	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q2	I found it easy to select the target	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q3	I feel that I performed quickly with this technique	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q4	If I had to use AR interfaces like this regularly, this is a technique I would appreciate having available	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q5	I found using this technique physically demanding	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q6	I found using this technique mentally demanding	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q7	I found this technique frustrating	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
	Please add any comments about the condition.			
	When you have finished all three conditions.... Rank the conditions 1-3: (1 = most preferred, 3 = least preferred)			

Post-Condition Questionnaire for High Density Conditions

Subject: _____

	With respect to this high-density condition…	**Direct Touch**	**Ray Casting**	**Magic Lens**
Q1	I found the selection technique easy to understand	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q2	I found it easy to select the target	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q3	I feel that I performed quickly with this technique	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q4	If I had to use AR interfaces like this regularly, this is a technique I would appreciate having available	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q5	I found using this technique physically demanding	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q6	I found using this technique mentally demanding	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
Q7	I found this technique frustrating	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree	☐ ☐ ☐ ☐ ☐ ☐ ☐ Disagree Agree
	Please add any comments about the condition.			
	When you have finished all three conditions…. Rank the conditions 1-3: (1 = most preferred, 3 = least preferred)			

Post-Experiment Questionnaire

Subject: _____

Overall Preference

Please provide an overall rank for the interfaces you used. Write a number (1-3) next to each, with 1 being most preferred and 3 being least preferred.

_____ Direct Touch

_____ Ray Casting

_____ Magic Lens

Try to list three things you liked about the interface you marked with a 1:

Try to list three things you disliked about the interface you marked with a 3:

Have you ever encountered a Magic Lens interface before?

☐ No, this was the first time.

☐ Yes, in: *Tick any that apply and provide details if you can.*

- ☐ A research application
- ☐ A commercial application
- ☐ an academic paper, textbook or website
- ☐ fiction (movies, TV, books...)
- ☐ something else?

Details:

Please add any other comments or feedback here: