Navigational System and Desktop Environment Design Within the Virtual Space

Adam D. Quirk B.Des (Graphic)

A thesis submitted in fulfilment of the requirements for the degree of

Master of Design

School of Design, Communication and IT

The University of Newcastle, Australia

November 2006

1 Declaration

I hereby certify that the work embodied in this thesis is the result of original research and has not been submitted for a higher degree to any other University or Institution.

(Signed): ....

2 Acknowledgments

I would like to thank my supervisor Michael Dickinson for his guidance, encouragement, and support throughout this research. As a friend, colleague and mentor he has continued to provide me with the appropriate push as well as the room to move, to allow me to complete this in my own way.

Thanks to Dr. Keith Russell, Cynthia Boyle, Anthony Nicholls, the late Sue Park, Roger Quinn and Allan Morse who each in their own way inspired, drove or challenged me to think, question and reconsider my viewpoint and direction during my undergraduate studies and more recently as colleagues.

Thanks to Chris Carr for his consistent backup, beerside discussions of the possible and impossible and for having the ability to explain the wonder of statistical analysis in ways my previous maths lecturers could learn from.

Special thanks to my parents and family for believing in me always. Even when you haven’t understood what I do or what I’m doing, your support has always been unfaltering. Special thanks to Mum and Dad for ensuring that education was always available to me, even when it meant sacrifice to you. Thanks to Barbara Singer for the academic inspiration.

Lastly, thank you to my wife Sarah. Your support and assistance has enabled me to complete this task. Your confidence has helped me maintain momentum and stopped me from deleting this tome on many occasions. Thanks and much love.

Contents

List of Figures

List of Tables

3 Abstract

This study explores the development of patterns for the visual design of interface elements within a virtual environment. The document will outline the process for this study and will formalise an approach for future research.

Commonly, existing interface systems allow for the representation of data storage, manipulation and navigation via two dimensional structures. With the emergence of virtual reality systems in medicine, military and entertainment there becomes a need to transform this limited two dimensional representation into one that best facilitates the new environment. The proposed model will incorporate application menu systems and the modes of manipulation of data in specific applications.

The framework for a prototype of the interface has been produced as well as a structure for assessment via user interaction and response. This framework will provide a template for understanding the base interaction with the operating system, that is, how to organise files and initiate software, as well as the operation of a simulated software package. If the prototype were built then it would be viewed as a virtual environment and the interaction could take place via the use of either mouse (or similar prop) or data-glove. Future study could include the actual building of each of the interface variations and putting the series of users through the prescribed experiment procedure.

Applications of an interface resulting from testing such as this could be found within a range of fields. Military training currently makes use of virtual simulation and this could provide access to information needed within operational procedures. Similar technologies could be adapted for pilot control systems within a visor display. For medical research, where virtual surgical techniques are being used, information could be accessed within the operating environment that had otherwise been contained in adjacent systems outside immediate reach. Real world application of these models would be limited at this time by the use of the appropriate computing power, however, the future use of this study could have broader application within the development of game technology, internet access and data mining.

The proposed experiment requires the development of interface variations based on a predefined pattern structure that informs the design of certain elements and of the tasks to be performed. The patterns included in this study form the initial set from which a library could be extended and developed upon. While these pattern definitions are crucial

to the execution of the experiment, they have been presented here in Appendix A to preserve flow and readability of the document.

The review of the current literature within this study covers a range of publications related to the development of virtual and augmented environment interfaces. A summation of the tools, devices, techniques and constraints which affect this area of development has been included. The review is presented to provide broad background information for the reader, building a context through which the experiment should be viewed.

The purpose of this study is to provide a method through which the visual representation of an interface can be more consistently assessed. This will be achieved through the use of a pattern language for a development framework, providing interface consistency in structure and principle. This is significant to the field as currently the assessment of visual representation is conducted across projects without a consistent framework and the subsequent learning is not readily transferred across applications.

4 Introduction

4.1 Motivation

Research into Virtual Environment (VE) interface and interaction has been undertaken since the 1960s when the concept was introduced (Sutherland, 1968). There have been a large number of individual components developed, usually for specific applications and developed on an ad hoc basis, suitable to specific application. Studies by Bowman on interaction techniques (Bowman, 1999) and usability (Gabbard, 1997) have helped to establish a common reference point for user interaction in VE.

By using the basis laid down by Bowman, Gabbard and other current research examples this document will move towards an assessment framework for visual representation of interaction use in virtual and augmented reality environments. The increase in computing power and associated technologies over the past few years has allowed for substantial advances in capabilities of VE systems and improved visual displays. One such technology is the improvement of Head Mounted Displays (HMD) to more wearable glasses. New developments using direct retina projection allow the equipment to be even further improved in an ergonomic sense and will provide many new applications for augmented reality (BBC-NEWS, 2004).

This study is a step towards to the development of guiding principles and processes for interface design for virtual and augmented reality experiences. Currently there is little information readily available on the actual visual representation of interface with these environments and the preference users have in relation to this interaction. There have been many discussions on the structure of interaction tools, such as the TULIP system (Bowman and Wingrave, 2001), and the recent development of virtual tools utilising 3D widgets of various constructions(Döllner and Hinrichs, 1998, Lindeman et al., 2001, Serra et al., 1999).

While 3D widget design has been shown to be effective in some scenarios, the representations are usually crude and the application varied. Lindeman et al (Lindeman et al., 2001) has shown that the preference, speed and accuracy of interaction was improved using 3D widgets coupled with physical props. This combination gave the user a greater sense of natural interaction and improved the level of comfort when performing specific tasks.

4.1.1 Context

The replication of real world process are not necessarily the best basis for designing interaction tasks (Bowman, 1999). Within the conventional 2D desktop environment deleting a file can be accomplished via dragging a file to a trash icon, selecting the file and keying delete or selecting the file and selecting delete from the pull down menu system. This provides redundancy which allows ease of use across a range of user types and highlights the base difference between efficiencies, the closest to the real world analogue being the least efficient and most complex. This poses several questions however for VR and AR, such as, how to include simple and accessible interaction with objects or system functions, how to visually represent these functions and how to initiate them.

In terms of task completion many tasks can be accomplished with variation on the 2D tools that many users are familiar with, such as drop down menus, buttons, sliders etc. These methods have been translated to VR and AR systems. While these have usually been constructed as replications of the 2D versions, they bring with them the need for a fixed frame of reference and the need to have a vaguely similar appearance across applications. The introduction of more specialised virtual tools will bring the need for a general frame of reference for a new set of evolving tools that have modes of interaction outside of the expected uni or bimodal tools currently utilised.

4.1.2 Augmented Reality Considerations

Recent advancements in AR have seen its introduction into commercial arenas previously removed from direct computer interaction. Applications such as HMD use in auto repair and mechanical analysis (BBC-NEWS, 2004) and adaptation by the military foot soldiers for in battle communication and status reporting. Other applications like the Tinmith Project (Avery et al., 2005), based in the Wearable Computer Lab, are investigating the possibilities this technology can provide, experimenting in environmental modelling and AR games.

Within AR the distinction between elements that are generated and those that are real will greatly effect how the user interacts with and accepts the interface. Elements that are distinctly fixed in the environment will clearly be virtual, however, their appearance will alter the way in which these enhance or detract from the overall impression of the application and uptake of user. Difficulties due to registration tolerance and the need to use real proportions of measurement within AR means the use of props as part of an input device will be a challenge which will need to be overcome.

4.1.3 Current Development and Technology

No general standard for interfaces exist and differing applications are currently being used. The modes of interaction currently in use fall into some broad categories; direct interaction, physical tools and virtual tools (Mine, 1995). From these we see experiments in the use of combinations of these interaction structures and the input methods available.

Users tend to prefer interaction with props, or at least feedback to indicate surfaces rather than no surface structures and only visual feedback (Lindeman et al., 2001).This factor is re-enforced whether the interaction is with 2D or 3D widget interfaces (Martens et al., 2004) and operates similarly in either VR or AR. It can also be seen that higher abstraction through 3D widgets can improve some tasks but is less direct than interacting with the real world (Wloka, 1995).

The user’s context brought to an environment is one consideration which should remain throughout the development of any interface system for VR or AR. The transition from the real world to the immersed or the expectation of methods of interaction with virtual object will depend greatly on the user’s level of experience and exposure to similar environments. Users may need to be forced, or at least encouraged, to explore possibilities offered by the paradigm shift from real to virtual. Pierce et al (Pierce et al., 1999) showed this when designing a virtual tea party. Users were positioned to encourage turning of the head and given real tools, a flashlight, which was transferred to a virtual analogue to encourage use of the direct manipulation of objects within the VR.

There have also been several advances in the physical tools of interaction that have progressed parallel to the computer and display evolution. The continued development of high and low end haptic devices is leading towards more manageable and comfortable methods of interaction. The use of vibratory feedback for example (Lindeman and Templeman, 2001) via simple use of rotary vibrators from mobile phones has proven to be a successful method of providing feedback. The mobile phone itself is moving towards a more convergent device itself with most new models now housing a range of computing, audio, visual and communication functions beyond the simple notion of the telephone. As a result a large portion of the population (first world at least) is now seasoned to carry around at least one digital device. This could be the introduction point for commonplace augmented reality devices and would be a logical step as the portability verse processing power divide decreases. While the development of such devices, the social and technological trends and the possible outcomes, are somewhat beyond the scope of this document the consideration of these possibilities provides a frame of mind, which presents a component of the motivation behind such a study.

As the reality of portable, wearable and ubiquitous computing becomes an accepted norm the need for common metaphors and modes of interaction emerge.

4.2 Problem Statement

Analysing the methods and processes of interaction tasks within virtual environments (VE) is a difficult process. This is due to the ad hoc nature of the systems which are currently in use and the project specific requirements that each interface and interaction structure has required. Recently, several studies in this area have focused on providing usability characteristics (Gabbard, 1997) and building taxonomies of interaction tasks (Bowman, 1999) based on the existing systems and going back to the basic studies conducted for the development of 2D interfaces (Foley, 1979).

Bowman broke the interaction tasks into four general categories; travel, selection, manipulation, and system control (Bowman, 1999). Travel being the movement of the user, or at least the user's viewpoint, or the actual traversal of the VE world. Selection is the process of the differentiation of one virtual object from another in order to interact with it, this definition is also appropriate for objects, navigational devices and control elements. Following on from selection is manipulation, which constitutes any action or process that affects the original state of the selected object, such as repositioning, change in scale or using a virtual tool to initiate some form of behaviour. The last of the general task categories is system control, which is the process through which the user can initiate changes in the system behaviour from within the VE. This may be as simple as saving a location or changing to a different VE application. For augmented reality systems, this interaction could be the opening of other applications or establishing a network connection.

Currently many studies (Gabbard, 1997, Lindeman et al., 2001, Wloka, 1995) are centred on the need to define the type of interactions executed for performing specific tasks. That is, attempting to pre-empt the types of interaction techniques which will be necessary to overcome the range of novel processes that this form of computing presents. There has been little work on the visual structure of these tools in the augmented space. Most iterations comprise of an interface built on known 2D representations or simplistic 3D objects as integrated by the developer of each experimental interface. While this serves well to test the nature of the interaction task it leaves the visual acceptance, robustness and user preferences aside.

Meaning, the visual nature of the user interface is being driven by the available technologies rather the user's needs. This document outlines the development of several user interfaces, utilising common visual paradigms and for the performance of a range of tasks drawing from those outlined in Bowman and Wingrave's four interaction categories that is, navigation, selection, manipulation and system control (Bowman and Wingrave, 2001). The interface variations will be defined by a general set of guiding structures or patterns describing the mechanical function with indicators as to preferred solutions for the visual representations. These definitions will be supported by further description of the specific visual variations of element treatment but will not be as prescriptive as to actually develop the interface designs themselves.

The overall aim of this process is to contribute to the development of universal standards for the use of graphic tools within virtual and augmented environments. As the WIMP model (Windows, Icons, Mouse and Pointers) has become the standard for 2D computing the visual nature of this interface has been redefined and explored many times over. The exploration of the three dimensional space is due and as with the current 2D space a recognised set of standards needs to be developed. That these standards are generally assumed or form part of a code of practice, is dependent on the development community and the users themselves.

The objective of this study is to explore a process and that will provide a testing and development baseline for virtual interface design. This set of patterns and the related testing process could become part of the standard development process for interface design. The framework enables the development of difference and the gradual tuning of known interaction metaphors and representations as the tools and the general user's knowledge progress.

4.3 Scope of the Research

Initial work by Quirk (Quirk, 2005, Quirk, 2003) investigated the development of user interfaces and possible structural development guidelines as part of a larger overview study of 3D virtual interfaces. Since this study began 3D User Interfaces: Theory and Practice (Bowman et al., 2004) which covers similar terrain was published. This publication focussed the scope of this study to investigating the visual design aspects of the 3D user interface elements. Bowman, Kruijff, LaViola and Poupyrev have completed broader studies in an effective way and indeed from a point of view at the forefront of the field.

This study has subsequently focused on a specific aspect, the actual visual representations of the interfaces which are constructed for virtual and augmented realities. However before it is possible to begin investigating the many variations of interface representation, it is crucial to have a framework for the assessment so that the differing representations can be assessed in a similar, replicable fashion. The tests outlined in this document are designed for a system with applications viewed in the first person, and makes use of a visual display and a physical input device.

With hardware and software is constantly undergoing change this document does not specify the application software which should be utilised, nor the hardware devices required.

Due to the limited access to the appropriate hardware and software this study has been restricted to a theoretical rather than practical realm. As a result a more dedicated study has been made on the types of systems and interactions currently in use with the view to future work which would be carried out on the appropriate equipment.

4.4 Hypothesis

This document provides a careful analysis of current and appropriate development methods and representations which will contribute to the prototyping of 3D interfaces through the use of pattern definitions to provide a conceptual framework for consistent interface testing of applications across multiple environments.

An experiment will be described where users will be presented a series of tasks within a VR or AR space with varied visual interfaces. Utilising a combination of 2D widgets with props and 3D widgets with direct manipulation, the presentation of these differing elements can be isolated and the benefits of the visual design measured.

The hypothesis, which underpins this study, is:

If through pattern definitions, 3D interfaces can be consistent in principle and structure, then the effects of variation in visual representation on the user’s ability to successfully perform set tasks can be more specifically assessed.

This hypothesis was developed during the initial stages of the literature review as it could be seen clearly that the visual representation of interface was lacking investigation and a repeatable capability for assessment would be required.

4.5 Overview of the Study

This study includes the motivation, a series of definitions of terms, the problem statement, document scope and hypotheses. This introduction provides an entry point for the reader and provides an overview to the context of the document and the process which surround its authoring. The following review of the literature covers the intellectual background to studies relating to the development and design of VR and AR interfaces and their visual representations.

The materials and methods of the experiment, detailed in chapter 6, explain the processes and tools used for the planning, analysis, design and assessment of the interface experiment. This section concludes with an explanation of the expected results and the possible variations, and conclusions which can be drawn from the study.

Following the conclusion is a set of appendices which document the conduct of the experiment and these are to be read in conjunction with this document. Appendix A contains the full explanation of the patterns for use in the experiment described. Following this, Appendix B provides a sample test which can be used during the experiment process, Appendix C provides a questionnaire for determining the selection of users and Appendix D provides the Post Task Evaluation. The pattern reference structure is considered a core component of this document though it has been located and the end of the main document section to preserve readability.

5 Review of the literature

The review of current literature was focussed towards answering six areas of inquiry. These areas of inquiry provide a basis for the knowledge required of the VR and AR environments from which reference can be drawn to complete the experiment design. The review is targeted on developing the background knowledge as required for the development of 3D interfaces through the use of pattern structures, with the view to assessing the nature of the visual representation of the interface elements.

The areas under investigation are limited to literature related to virtual environments and augmented reality. The following research questions have been used to focus the review.

1) What is navigation? Historically, mechanically and virtually.
2) What is interaction? Historically, mechanically and virtually.
3) What are the key interface elements required? That is the generic verses the specific, and how does this relate to system access, task based interactions and navigation.
4) How will these be used? The usability and ergonomics of manipulation and the use of haptic, aural, static and dynamic elements and there associated interaction techniques.
5) How will these be represented both graphically and mechanically? The representation of the tools and interface elements.
6) How will this be applied in a broad sense? What are the possible uses and specific applications of these kinds of tools?

This review and its structure provide background information from which to move into the experiment. This review seeks to build the appropriate level of understanding required for the construction, development and testing of the interfaces as described in the experiment. This section will end with a brief summary of the key findings which will carry through to the experiment design.

5.1 Navigation

The process by which people control their movements and locate objects within any space is referred to as navigation. Two essential processes, form the basis of navigation within the immersive environment; cognitive mapping and spatial ability. “A major problem for users in all forms of navigation is how to maintain knowledge of their location and orientation while moving through space (Ben Hajji and Dybner, 1999).”

Cognitive mapping is the process of developing an overall relationship of separate elements within a system or environment. The user will develop, through contact, a set of identifiers that allow for the understanding of placement and orientation of specific elements and the process of moving between these. The ability to do this is greatly increased by the addition of stronger cues. These cues may include paths, edges, landmarks, nodes and districts (Darken and Sibert, 1993, Ingram and Benford, 1995).

Spatial ability is usually broken down into three areas; orientation, visualisation and relations. That is, the ability to move or transform the stimuli or object while retaining relationships, manipulating relationships within an object and an understanding of how the object would appear from a differing perspective (Ben Hajji and Dybner, 1999). Catering to the range of spatial ability is a key challenge for interface design and task performance can be improved through the use on compensatory techniques for the users with low-end spatial ability (Chen et al., 2000).

5.1.1 Navigation Types

Techniques of navigation can be subdivided into two core components; travel (motor) and wayfinding (cognitive) (Bowman, 2002). Within these two components are a series of navigational subtasks. These routines describe the common tasks executed when either executing movement within the environment or building an understanding of the environment itself.

The variations of navigational tasks can be placed under three broad categories:

1. **Exploration;** no explicit target, the user is investigating the environment and available objects.

2. **Search;** specific target, the user is directly seeking a particular place or object.

3. **Manoeuvring;** short range, high precision (performing), usually performed within the execution of specific tasks

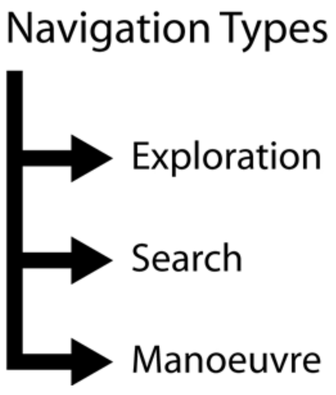

Figure 1. Breakdown of navigation.

5.1.2 Navigation Techniques – Travel

As in the real world, travel is conceptually simple in that it is the movement from one place to another, within immersive VEs this is the movement of the viewpoint rather than the user. The notion of travel can be subdivided; there are five common metaphors for travel interaction and these are best summarised by Bowman, Kruiff et al (2001);

1. **Physical movement;** user handled
2. **Manual viewpoint manipulation;** user handled
3. **Steering** (most common), gaze directed etc; user controlled system handled
4. **Target based;** system handled
5. **Route planning;** user planned system handled.

Physical movement is used when there is a need for specific physical exertion when travelling. This is used when an enhanced sense of presence is required and may involve the use of devices such us stationary bicycles etc.

Manual manipulation involves the physical motion of the hands to control the travel, such as grabbing the air and pulling forward.

As the most common technique, steering is simple and effective. It is executed by simply gazing or pointing in the required direction and initiating the movement via secondary command. The direction is controlled by the user as the system handles the actual movement.

The simplest form from the user perspective is target-based navigation as the user only has to specify the destination and the system handles the transition. This can be instantaneous, as in teleportation or via some transition between points.

Finally, route planning is very similar to target based travel only here the user specifies the path between points, specifying multiple targets. The system takes control of the transition.

5.1.3 Navigation Techniques - Wayfinding

“Wayfinding, the counterpart of travel, can be described as the cognitive process of defining a path through an environment, thereby using and acquiring spatial knowledge to build up a cognitive map of an environment (Bowman et al., 2001).”

There are two major components to this process;

1. **User centred;** visual motion, large field of view, vestibular (real motion) & audio
2. **Environment controlled**;

   2a. Structural organisation; definition and relation of parts

   2b. Cues; real world principles, maps, grids, architectural, mood etc.

The visual design of the interface elements effects the user’s wayfinding processes by highlighting or subduing the differing cues. The representation presents difficulties for the wayfinding process when the representations are inaccurate or incomplete. The use of visual cues and organisation can be re-enforced by haptic and auditory cues which assist users, especially those with visual impairments to form the appropriate cognitive maps (Semwal, 2001). The notion of legibility in virtual spaces is being explored in a similar way to city planning. That is, the development of clear patterns of structure which allows users to develop the cognitive mappings which improve the wayfinding process (Ingram and Benford, 1995).

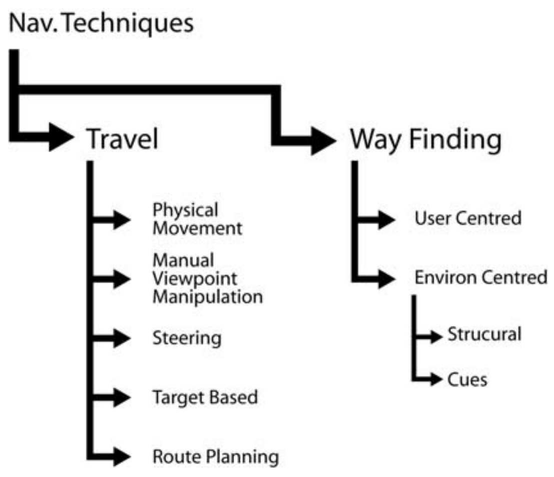

Figure 2. Breakdown of navigation techniques

5.2 Interaction

Interaction refers to any process or method through which a user performs a task via a user interface. There are a variety of techniques that are currently employed within VEs, these are generally based around the methods used for selection and manipulation.

5.2.1 Selection

The two main types of selection are ray-based and reaching techniques with a third, which is multi-modal (Mine, 1995).

Ray-based is any technique where the user specifies a particular object for selection by pointing to it with a ray, like a laser pointer, via some method or another. Some employ a direct ray extending from the finger others use a combination of eye and hand gestures or gaze-directed selection.

The second common selection technique is reaching, which is modelled more directly from the user's normal processes. Put simply if you want something, you reach for it grab it, as has been liken to the 2D drag and drop process (Steinicke and Hinrichsy, 2006). This method is complicated by scale, and works best for objects close to the user. Distant objects involve the scaling of the world or hand, which can be problematic.

Thirdly are the multi-modal techniques which combine serval types of human interaction. By combining text, speech, position, gestures, eye movement etc. complex interactions can be accomplished. These interactions are more useable than if one interaction was executed alone (Wingrave, 2001, LaViola, 1999).

5.2.2 Manipulation

The process of manipulation can be defined as; "the action of touching with the hands (or the skilful use of the hands) or by the use of mechanical means" (Cognitive Science Laboratory, 2006)

This could not be more fitting in the computer sphere than within the arena of virtual or augmented realities were the manipulation can be both the action of touching with the hands and via a mechanical means simultaneously.

Common methods of manipulation include simple selection and repositioning, scaling and rotating and generally altering the properties of a represented object. This may or may not also represent an action that occurs in an adjacent system or the real world that should indicate a point of difference for the visual representation.

Interactions which lead to manipulation of objects or the environment can be represented in a variety of fashions and should attempt to convey a sense of function in their representation. While this does not imply that the representations need to be pictorial, the representation should aid in the recognition and at best be functional in themselves. An example would be vertex handles used in 3D widgets which imply a point at which manipulation can be exerted (Döllner and Hinrichs, 1998). When the user grabs a handle, the mechanics of available interaction could be easily displayed via arrows indicating the ability to scale or rotate.

In general, manipulation should be represented as a separate process to the selection component of an interaction and should make use, where possible, of known techniques. This reduces the complexity for the user and provides obvious function when the types of available manipulation are not easily apparent.

Many manipulation techniques from the 2D environment can be translated to the 3D-user environment though these should only be used only where appropriate. A 2D slider works well for adjusting a range of data types and users are already familiar with the method of use. This may not have the level of accuracy of say keyboard input, but is suited to input methods where relatively precise input is required but the access to additional hardware may be cumbersome for the system.

Studies investigating the use of props in conjunction with manipulation tasks have shown that users perform better with the use of props and at minimum simulated surfaces (Lindeman et al., 2001). Lindeman and his team also felt that the use of 3D widgets over 2D increased the accuracy of the user’s interactions.

5.2.3 TULIP

One of several types of input devices that are commonly used within VEs are pinch gloves. These flexible gloves house sensors that monitor the movement of the hands and fingers and provide feedback when, in particular, fingers are pinched together.

Bowman and Wingrave developed a menu system known as TULIP or Three-Up, Labels in Palm based around the flexibility offered by pinch gloves (Bowman and Wingrave, 2001). The system makes use of the virtual hands and fingers by placing menu items on each finger and additional menu items on the palm of the hands. The user then can make selections based on pinches using the appropriate digit. The little finger cycling the long menu items from the palm to the fingers so that direct selection can be made.

5.2.4 Tablet

The use of a stylus and tablet for menu selections is a common function within VEs. A tablet, consisting of a flat surface, is physically held by the user and represented as a plane, with the menu clearly laid out, within the virtual environment. Selections from the displayed menu are made via a stylus that is held by the user and represented by a pointer in the VE. This method allows the user to see all the menu options at a glance and quickly select from the options available (Serra et al., 1999). While this represents the quickest way for users to see all available options and make selections it is not always practical, as it requires a larger amount of space to represent the information.

5.2.5 Floating Menus

Floating menus are the most similar to the pull-down menus used widely in 2D desktop environments. The menu options are categorised under broader headings and the subcategories and nested below. Initially the headings are displayed and upon selection reveal the available options allowing each option to then be selected via a pointer, just as we do with a mouse within the WIMP model. Within virtual environments the upper level headings are attached in space to the user’s head, so from the user perspective they occupy a fixed position. This has advantages in that multiple options can be nested while taking up minimal space, however, it can cause visual obstruction of other elements within the VE.

5.3 System Control

System control generally refers to any task where the user applies a command that changes the system state or mode of interaction (Bowman et al., 2001). Little has been developed so far with respect to system control within immersive environments and as Hubbold presents it, this is due to the difficulty of delineating the system from the application (Hubbold et al., 1993). Most work that has been completed is ad-hoc, still trying to remove itself from the models of the 2D world and the interaction modes that are familiar within it. A common example of this is WIMP (Windows, Icons, Mouse and Pointers). Complexity of input data to control elements within the virtual space based on the number of points required to describe an object and the number of degrees of freedom that input devices require has limited control to the 2D model. Overall these tasks are often integrated into other interaction tasks and ideally this integration is a seamless or “Modeless” interaction so the user can retain focus on the main task.

There are four base groups of interaction that govern system control:

1. Graphical menus
2. Voice commands
3. Gestural Commands
4. Tools: virtual objects with specific application

5.3.1 Graphical Menus

Graphical menus are the most familiar form of controls and borrow from the models formed in 2D interaction. Since these models are commonly understood it allows for fast uptake by users when presented within VEs.

5.3.2 Voice Commands

Voice commands can consist of multi-level functions or menus that are accessed via speech. These commands offer great flexibility but are difficult to learn and require the user to memorise the functions available. Voice commands are often used in conjunction with other interaction modes to increase the flexibility of the command (Billinghurst and Savage, 1996).

5.3.3 Gestural Commands

This form of command is most often controlled with the hands and makes use of data gloves or similar input devices. The user makes a specific gesture, such as pinching or pointing, and this triggers a predetermined event. One example of this is the TULIP system. LaViola (LaViola, 1999) discusses these techniques used in conjunction with speech input. This method can be extremely flexible but can induce fatigue in the user if the system operates under certain conditions due to the amount of physical movement required to execute some commands. While gesture driven interfaces have the potential to provide complex, intuitive and flexible interfaces, the complexity of interpreting the set of gestures makes this type of interface difficult to achieve (Wingrave, 2001). To fully realise the potential offered through gestures systems will need to recognise sequences of hand postures, which may equal more than the parts, and be able to adapt on the fly to accommodate personal nuances rather than simple rote learnt commands (Gabbard, 1997).

5.3.4 Tools

The integration of virtual tools into VEs allows for complex objects which mirror natural devices or perform “Magic” applications. The use of tools allows the system to offer specific commands in a recognisable format that can reduce the complexity of interaction and increase the complexity of the actions generated by specific events. Tools are often positioned like graphical menus and occupy a fixed position, like a compass or map that can be selected to direct the user to new locations.

5.4 Generic and Specific Interface Elements

Approaching the interface from a visual design perspective, we can breakdown the requirements into key elements dissected by function. The core areas of function, which would invite specific treatment, are system access, navigation and task based activities.

5.4.1 System Access

System access and control is usually treated as a baseline across multiple applications and provides a point of interface consistency. This allows users to gain orientation with the broader system rather than application specific requirements and provides a start point for the investigation of the application's specific functions.

In a 2D desktop environment, system access is based around the same mechanical interaction technique as most specific application functions, the WIMP model. To further consolidate the user's previous experience, common dialogs, prompts, menu locations and key commands are used for the majority of applications running under a specific operating system. This allows users to quickly familiarise themselves with the areas that form the system functions of an application and to differentiate the application specific tasks. If the system controls were to continually be accessed via varied methods then the cognitive load on the user increases dramatically for each new application.

5.4.2 Navigational

The two main processes for navigation, travel and wayfinding are catered for in most virtual or augmented environments in either a novel fashion (mainly VRs) or mirroring 2D environments (mostly AR). These methods could be refined into base common elements for navigating the system level details and common task specific tools for interacting with the specific application process.

This can be achieved in a variety of ways and has been attempted with tools such as the world in miniature method (Stoakley et al., 1995). This allows the user to see a scaled down representation of the complete environment in their hand, with the ability to move around from that representation. This is suitable when the need is to manoeuvre around a large environment but it not necessarily the best metaphor when viewing a complex object and looking to adjust the viewer's position relative to that object, a building or molecular structure for instance.

For the purpose of this document, the virtual space will not represent a large-scale environment and will be restricted to a representation of a desktop like application or task. As such, the notion of travel is limited to the positioning of the user in the environment to allow for better observation of the objects present rather than to explore the world at large. This limit is also more closely aligned with the needs of augmented reality applications where the user is consistently grounded by the real world points of reference but may need to manoeuvre themselves around virtual objects.

5.4.3 Task Based

The representation of task based interface elements is by nature as varied as the tasks to be performed. The objective therefore is to find a higher level series of metaphors or representation types to allow for an expected interaction with each novel interface element. This can be most easily seen in the nature of a button, users of nearly all applications which use a Graphical User Interface (GUI) will recognize a three dimensional object with a label to be a functioning element. Usually this object is labelled clearly with the outcome of the interaction it suggests, such as “Delete” or “Save”.

This obvious representation model becomes more difficult as the degrees of freedom of interaction increase and the visual representations become more complex. The introduction of new common metaphor elements will be required for the new task types which are becoming available through the use of VR and AR environments. These task types may include a mix of simulated real world interactions, which would require direct translation of the real world interface and more involved virtual only interactions such as zoom, stretch or search network which will require specific representations within the virtual space.

The current set of 2D interface elements which are commonly expected when performing task based operations include; buttons, sliders, menus, dials, multi-state buttons, input dialogs and vertex handles. Many of these could be translated directly into the virtual space. Each mechanic could be represented in its own specific way, relative to the application present and without much education for the user to be fully functional.

5.5 How Will These Elements Be Used?

There are a large range of interaction devices available and there is a consistent thread throughout the development of computing to investigate new devices as new paradigms of computing are developed. These range from the, now common, keyboard and mouse through to voice command and full body movement based interaction devices.

In this section a limited set of differing interaction devices will be explored. These may require differing visual representations, or be used in conjunction with a visual representation.

5.5.1 Visual Display Types

Since Sutherland (Sutherland, 1968) put forward the head mounted three dimensional display, developers have dreamed of light weight, high resolution wearable display devices. However, the continued development of 3D display type has revealed that this device is not the most applicable in all applications. As such there has been a range of display types developed, each with their own advantages and disadvantages. While it is not the purpose of this document to review these, it is worth noting the common types and their general usage, as this may affect the type of visual representation applied to interface elements.

A solid discussion of the general visual display devices can be found in Bowman et al's 3D User Interfaces (Bowman et al., 2004). These can be summarised as:

Monitors

Conventional CRT monitors are used in conjunction with stereo glasses to produce stereopsis. This combination uses one of several techniques, that is shuttering, polarisation multiplexing or spectral multiplexing, to provide one image to the left eye and another to the right thus creating the illusion of 3D.

Key advantages: The lower cost and the access to a full range of input devices as the user is usually sitting at desk with keyboard and mouse available.

Key disadvantages: The small field of regard (FOR) due to the size of the view space and positioning and the fact that if the user moves or interacts in front of the monitor the 3D illusion is broken.

Surround Screen Displays

Surround screen displays are generally large screen, projection based displays where the user can move around within the display. The screens are typically rear projected so that the user can move about freely without casting shadows on the projection. The first such system was the CAVE (Cruz-Niera et al., 1993) consisting of three side and floor projection. Since then there have been numerous variations of various scales from the personal to large reconfigurable systems such as the reFlex system by FakeSpace Systems (FakeSpace Systems, 2006). This example can be connected to provide large multi-person environments. Stereopsis is produced in the same manner as for monitor projection and therefore has similar issues with depth and occlusion.

Key advantages: High sense of immersion, multiple users can view the display. The system has a large FOR and field of view (FOV) that is the area on which visualisation is available.

Key disadvantages: As there are multiple displays this configuration is computing intensive. Single view point only when multiple users are present.

Workbenches

Available in various configurations from a variety of manufacturers Responsive Workbenches or just Workbenches are projection based displays which are based on the need for interaction tasks which would normally occur on a specific surface such as a desk or bench. These are usually configured as either a flat panel which is vertical, horizontal or inclined or a combination in an L-shaped configuration. As with monitors the FOV and FOR is restricted by the size of the display and the viewing aspect.

Key advantages: Good visual quality due to size, high spatial resolution. Good for direct selection techniques as the objects are within arms reach. Can accommodate multiple users but with single viewpoint.

Key disadvantages: Not as suitable for multiple view points. The systems are limited by physical scale. Can accommodate multiple users but with restriction like surround screen displays on the individuals perspective of the environment.

Hemispherical Displays

With 180 by 180 degree FOV hemispherical displays offer a brighter image due to their front projection. The systems are available from a smaller personal system through to large multi-user environments with similar restrictions to monitors and surround screens in terms of view point restriction. These displays are characterised by having a hemispherical dome placed in front of a projector with a wide angle lens. The image is pre-distorted via software and provides a bright image though not uniform across the display with higher quality and resolution towards the screens edge.

Key advantages: Bright image, input via keyboard and mouse or other familiar tools.

Key disadvantages: Loss of quality across the display. There are issues with shadow casting and occlusion of virtual objects if user nears the screen.

Head Mounted Displays

One of the common head coupled displays is the Head Mounted Display (HMD). While there are varied constructions all essentially place the display in front of the user's eyes using small screens. These systems create stereopsis by rendering two separate images on two separate screens, rather than the sequential switching or colouring of projection based systems. The user has full 360 degree FOR due to the complete immersion and covering of the headset. The size of the actual screens however, causes the HMDs to have a small FOV which can reduce the experience. Some versions use a mounted camera to show through the real world or are see through, these are used in AR and mixed reality environments. HMDs limit the input devices applicable as users may not be able to see the tools and may cause motion sickness (McGee, 1998).

Key advantages: User has full 360 degree FOR. Multiple viewpoints, one per user. Brighter and more portable.

Key disadvantages: The hardware powering the system must be capable of rendering to two channels. The small FOV can reduce the experience. Lower resolution than projected displays due to screen size and weight considerations. One size may not fit all.

Arm Mounted Displays

Available in several configurations, the arm mounted displays are simply a display device attached to a central armature. The screen is counter balanced to improved the ability to easily rotate the display around it's centre and allowing for the smooth shifting of viewpoints. Similar to the HMD in terms of visual cues and as a result has many of the same issues.

Key advantages: The ability for more than one user to quickly observe. High quality hardware can be used as there is less weight restriction.

Key disadvantages: User must constantly have contact with the display to position it, reducing interaction options. Limited field of movement due to the armature mechanic.

Virtual Retinal Displays

Also called light scanning displays, these displays work on the basis of displaying images directly on the retina. Controlled light, laser, is used to draw the image directly and use either three colours, red, green and blue or monochromatic versions with just one colour. The image can be shown as either fully immersive or see through catering to different use modes. With a FOV nearing human sight and bright, high resolution, stereo images available these displays have a large potential for a large range of applications. One deficiency is the lack of eye tracking, as the movement of the user's eye can affect the image, this is currently being investigated by the addition of eye tracking tools.

Key advantages: High FOV, fully immersive FOR available, multiple modes available, high quality imagery.

Key disadvantages: Eye tracking issues, visual cues issues due to consistent focus.

Autostereoscopic Displays

These displays generate 3D imagery without the use of polarised or shuttered glasses. This is completed through a variety of methods though mainly lenticular, volumetric or holographic techniques (a full examination of which is beyond this document). Lenticular displays use either vertical gratings or a cylindrical lens array which sits in front of the display providing different versions of the image to each eye. This is only effective when the user is positioned correctly. Volumetric and holographic displays both generate “true” 3D images by actually illuminating points in 3D space. Both types operate within closed environments restricting the size and space available however because of the “True” nature of the 3D image the image has essentially unlimited viewpoints.

Key advantages: User is free from extra hardware. Some provide essentially unlimited viewpoints and near true 3D images.

Key disadvantages: Users must be in correct position for best use for lenticular displays. The displays have limited volume therefore not suitable to immersive applications.

5.5.2 Haptic Devices

There is a large range of device types used for input or display which rely on physical response and feedback, these are collectively called haptic devices. These devices rely on either tactile cues (texture, temperature or pressure) or kinesthetic cues (position and movement). Much work is being conducted in the development of such systems and they will no doubt play a role in the development of the types of visual interfaces developed.

Haptic display devices in many ways can be used as replacements for visual display information and can be used separately for displaying certain contextual information. For the purposes of this document the use of haptic devices is limited to the use of feedback which is coupled to the visual cues and interface elements.

The use of vibrotactile feedback in simple user interactions can greatly increase the sense of interaction and has been completed using simple tactors such as mobile phone vibrators (Lindeman et al., 2002) and exist in popular game consoles such as Sony’s PlayStation and Microsoft’s Xbox.

A very good overview of the differing types of haptic devices available and the pros and cons of their use can be found in the book 3D User Interface Theory and Practice by Bowman et al (Bowman et al., 2004). This references the range of haptic device types and presents a simple list of haptic display characteristics as:

- Haptic display characteristics: the type of output which may be tactile, kinesthetic or both.
- Resolution: spatial (proximity) and temporal (refresh rate)
- Ergonomics:

The two of these which closest relate to the visual treatment of 3D user interface elements are the resolution and ergonomics. The usability of the visual interface will be strongly coupled to the ergonomics of the device, while the actioning of visual elements is matched with the proximity, either activated or in stasis. The visual cues are reliant on the refresh rate of the display system to give accurate interaction responses.

5.5.3 Auditory Elements

The use of auditory feedback and interface elements is a familiar process in 2D interfaces and can take further steps to inform the user within the 3D space. The ability for users to obtain a sense of presence from a series of auditory cues (Warren, 2002, Barrass, 1998, Mynatt, 1995) is an area that is currently receiving attention from various researchers. This exploration has specific implications for users with visual impairments and this is the driving force behind the development of auditory only interfaces, though this is beyond the scope of this work. The use of auditory interface elements within a visual interface system is of more relevance.

The introduction of auditory information into three dimensional environments comes with a range of considerations. Funkhouser (Funkhouser et al., 2002) gives a solid overview of the necessary computational requirements and methods, modes and treatments of 3D sound usage. From this work we can see there are many different considerations when attempting to construct a synthetic version of a real world environment, and list some further considerations for immersive VRs as:

- System Latency and Update Rates
- Listening-point and viewpoint consistency
- User tracking and adaptive systems
- Loudspeakers arrays and wall reflections

The notion of “Sensory Substitution” is a commonly used expression for the replacement of one sensory input for another. As stated by Meijer (Meijer, 2006) the early sensory substitution uses included the Braille system for reading and has moved on to accommodate a range of implementations usually with goal of replacing missing or damaged sensory information with the input of a working sense. This same process however can be used as a replacement or enhancement for sensory perceptions within VRs.

Lecuyer et al put several feedback methods to the test including an auditory alarm to guide users through a workbench insertion task (Lecuyer et al., 2002). They found the “the auditory feedback is considered as the most efficient alarm signal.”

This supports the current use of auditory cues in 2D interactions, commonly accepted as increasing the engagement for the user. These cues can be as simple as the click of a button when depressed or auditory cues indicating the change of page (location) within a multimedia screen. This process can be expanded from simple cues like button clicks, to indicators of friction or movement, to full-scale representations of the data to be displayed (Meijer, 1992).

5.5.4 Static Verses Dynamic

Visual representations of actionable and informative interface elements in any interface can take many forms, from simple text links through to video streaming with layered information.

For the majority of interface elements which represent actionable elements users now tend to expect some level of dynamic visual representation when they interact with this element. This may be as simple as an underline when hovering on a hyperlink or an animated menu item in a video game.

The shift from static interface elements through to dynamic elements is a sliding scale. The framing objects which provide visual boundaries for areas of interaction or information display are the most static, moving through to the points of highest interaction which are usually the most visually complex.

5.5.5 Usability Considerations

Design and visual representation of new interface elements, to some degree, must take into account existing user knowledge and the expected metaphors. While this may not be a direct translation of existing visual structure or technique it should try to maximize the user’s ability to reuse what is known, lowering the burden of learning the new interface elements.

Usability is an important field, however it is impossible to attempt to address the full gamut of difficulties and complexities of usability as it relates to virtual system and interfaces in general within this document. The principles of usability will however be taken into consideration throughout the development of the proposed experiment and the planning associated with its delivery.

A breakdown of usability characteristics is drawn up by Hussey (Hussey et al., 2001) and from Mahemoff and Johnson (Mahemoff and Johnston, 1998b) which will be discussed below. This can be seen in high level terms as:

- Task Efficiency: Help users of varied experience levels to minimise their effort to perform their tasks.
- Reuse: Ensure that users can use existing effort and knowledge. Knowledge reuse is usually achieved via consistent look-and-feel. Reuse of effort requires mechanisms for saving and retrieving previous work.
- User-Computer Communication: Facilitate collaboration between humans and computers by appropriately representing changes to the system (which have been instigated by humans or computers).
- Robustness: Minimise the likelihood of users misperforming tasks and facilitate recovery from errors when they do occur.
- Flexibility: Account for users of different backgrounds and experience by allowing for multiple ways to perform tasks and by enabling user customisations of the system.

These types of definitions have been made by other authors but are outside the needs of this document. The overriding principle here is that the use of a commonly defined set of expected criteria is met with each component or element of the interface. This can be achieved through the use of design patterns, of which these categories mark the structure.

5.5.6 Patterns

Mahemoff and Johnston (Mahemoff and Johnston, 1998b) put forward the notion that as software developers have had ongoing successes with the transfer of knowledge, the consistency of build and the development of knowledge bases through the use of design patterns, so to can usability design. With this ability designers, information architects, developers and analysts can all measure the effectiveness and appropriateness of a particular design against and agree common set of goals.

The use of patterns was originally put forward by Alexander et al (Alexander et al., 1977) when defining known structures and processes for developing architectural and town planning structures, the process was then applied to Object Orientated Programming techniques and knowledge as detailed in Gamma (Gamma et al., 1995). From this point the use of patterns and pattern languages to transfer knowledge and solutions to common problems has ranged through different subject matter and in varying forms.

The use of patterns within this project is extremely useful for maintaining consistency across the varied interface designs while altering the visual framework, style and approach. By using patterns, as defined in Section 6.3.1 Pattern Definitions, a baseline can be defined for tasks, users, user interface elements and the system within which the tests takes place.

5.6 How Will The Elements Be Represented?

This section will look at the methods used and considered for the development of the interface variations used in this study. The following areas will describe with respect to the development of varied interfaces for similar tasks as required by the experiment, the graphic model used for construction, the use of audio and physical tools and the use of visual feedback. The section will conclude with some discussion of the types of applications to which these interface could and would apply.

5.6.1 Graphic Models

There are essentially limitless possibilities for the ways in which a graphical user interface can be represented, even when these interfaces are derived from the same set of goals and requirements or patterns.

Irrespective of the process of informing the needs of the interface design, be it a pattern language, a strict process of instruction or a general evolution from previously developed iterations each design will follow a structure which will take form from a basic graphical model. This model will aide the way in which we cognitively map this structure and the internal elements.

This graphical model structure was broken down clearly by Mynatt when investigating the interface in order to translate the graphical to the auditory (Mynatt, 1995). The relevant components of the interface are:

- Objects – that is the base elements to interact with.
- The attributes of the objects – The visual attributes which are common to the differing types of objects such as size, orientation, position and colour and so on.
- The affordances of the objects – this is the visual cues which imply functionality.
- The relationships between objects – this refers to the physical relationships such as proximity and the functional action associations derived through interaction.
- The names of objects – consistent naming allows for the transfer of knowledge between differing interface types for similar elements and functions.

The differing components described here can be utilised when specifying the differences in the interfaces which will be presented for the experiment below. Through coupling this information with the direction provided in the pattern descriptions it will be possible to predefine a range of similar functioning yet visually alternate interfaces which will be suitable for comparison.

5.6.2 Audio Properties

Barrass (Barrass, 1998) investigates the nature of sonification and reduces the definition to a two part structure when related to information; the requirements and the representation.

To fulfil the needs of sonification in terms of requirements, the needs are further broken down into three activity based areas which can be addressed. These are interpreting, understanding or communicating. This allows the designer to focus on the specific task rather than the environment as a whole and reduces the complexity of attempting to develop the correct generic response. This narrowing of focus allows for a better treatment of the requirements of an information processing task.

The representation is treated much like the structures used for graphic representations where abstracts, simulations and metaphors are used to provide perceptual devices which the user can relate to in order to make sense of their environment.

Auditory Cues – Earcons

There are many methods which allow the addition of auditory cues, these range from simple feedback elements, object based audio to the stronger use of directed auditory information through the use of tools like earcons.

"Earcons are messages that are created from auditory tones. The properties of earcons are generally pitch, amplitude, frequency, duration, tone, and special effects that may be delivered to musical tones." (Warren, 2002)

Through the use of this type of auditory messaging users can be informed, alerted or reassured in relation to the current task. Warren describes that while earcons and icons (graphic) have similar function and expected behaviour there are differences in their use. While both convey information, earcons are much more complex to understand when represented simultaneously than icons and it has been shown that vocal messages are more easily understood.

Environmental Audio and Ambience

As with multimedia development and game environment design a richer sense of immersion is possible with the use of ambient or environmental audio. The inclusion of traffic noise in a city scene instantly increases the user's sense of placement. While this effect is desirable in an immersive and possibly entertaining scenario, it can hamper the ability to perform tasks which rely on auditory cues and feedback. The clarity of sound, and its differentiation from the surrounding or background noise, should be considered in a similar sense to the use of contrast in elements for the visual definition of actionable screen elements. That is actionable elements should be distinguishable from those elements which have no actionable function.

Feedback

As with the development of graphical interface structures, auditory feedback can greatly enhance the user experience. There have been many studies into the nature of feedback and human responses to it as it is one of the key areas within usability studies (Meijer, 1992, Warren, 2002, Everett et al., 1998, Gabbard, 1997). The nature of this has flowed into the study of virtual environments as a result.

Testing a workbench insertion task and a range of feedback types Lecuyer et al found the Auditory alarms were considered and measured the most effective feedback over various visual and haptic feedback styles.(Lecuyer et al., 2002)

5.6.3 Physical Tools

A number of studies, such as Lindeman's (Lindeman et al., 1999a) interaction tests, have been conducted where the subjects are given physical tools which interact with the virtual environment. The advantages to this are the benefits of passive haptic feedback without complex apparatus and the ability to have the real world like experience of interacting with the physical objects. Other studies such as the Tinmith Project (Avery et al., 2005) use this combination of physical and virtual tools further within and AR, allowing users to see the physical tool and its associated virtual object.

Similarity to real world is achieved through this process only if the correct visual registration is available (Azuma, 1997). Once this has been overcome however, the combination of virtual and physical tools will have the benefits of the multipurpose or mode tool which can be delivered from one physical object. That is, a singular tool which varies itself contextually depending on the surrounding objects or environmental requirements. This kind of thinking has already been applied to PDA interfaces, some which double as mobile phones, remote control devices, calculators and media players. These have the advantage of touch screen interfaces which totally reconfigure for each specific application, including variation in screen orientation, interaction area and device connectivity. This same thinking could be applied to the development of 2D or 3D widgets attached to a surface interface area of a physical prop or tablet.

5.6.4 Visual Properties

Lessons can be drawn from the 2D environment when looking at the needs of the properties of visual elements. When completing an overhaul of its well known and recorded interface Apple released the Human Interface Guidelines to assist developers in the transition and to inform newcomers (Apple Computer Inc., 2006). Several key components to the visual properties of the interface are covered, although specific to the Apple OS in that document themes are repeated continually elsewhere, some of these which are relevant to this study and virtual interfaces in general are discussed below.

Feedback and Communication

Simply alerting the user when an error has occurred is not enough. Users should be kept informed of the processes their actions have initiated and better still, be forewarned. This can be accomplished through simple visual clues as to the nature of the action such as progress bars while loading large data sets or transitional graphics to show a process is underway.

Visually, feedback can be a simple inclusion of indicators of action applied to objects or elements which may be actionable or available for selection. This is often seen in the use of hover states which subtly shift the appearance or highlight an object when the user is positioned to activate the object by some form of selection or other interaction technique.

Visual feedback as a manipulation aid can be included as general system wide operations. This can greatly increase the ability of users to successfully complete complex tasks as found by Noma (Noma et al., 1996) and Lecuyer (Lecuyer et al., 2002), especially when associated with haptic or auditory feedback.

Consistency

Consistency is possibly the largest challenge for VR and AR interfaces as the user's environmental context varies so heavily from one application to the next. User expectations are established through the gradual exposure to the current and alternate systems and carries with it the expectations set by the outside influences such as previous computer usage and cultural bias.

The primary aim for any virtual interface is to at least be consistent within itself. This allows users to establish learned behaviour, anticipate the resulting actions of interactions and reuse learning. Consistency overall leads to user comfort and has long been seen as a fundamental of 2D interface design.

Perceived Stability

Related to visual consistency is the notion of perceived stability. This is the inclusion of the expected building blocks of the interface and the control of elements as expected by the user. Stability can be implied in each interaction by ensuring that visual registration is consistent in the virtual and physical worlds and by ensuring that objects which are or look actionable, function correctly. To put it simply, if an object appears like a button it should act like one.

Aesthetic Integrity

It is important to follow general design principles for screen based design. While defining these principles is outside the scope of this document the area is well documented elsewhere and is a constantly evolving field, recorded in publications such as the web design related examples by Veen and Nielsen (Veen, 2000, Nielsen, 2000). Areas for consideration for aesthetic integrity are covered here as other points in this section, in addition simple visual design principles can contribute greatly, such as:

- Consistent use of a small set of fonts
- Use a constrained colour palette
- Use consistent functional elements for similar functions

- Use consistent placement of similar functional elements
- Cluster functional components by function

While this is in no way a definitive listing, these few base guidelines indicate how we improve existing developmental interfaces and provide the basis for most of the accepted and widely used interfaces today.

Modelessness

While captured and controlled via system design, the principle of modelessness can be inferred and reinforced through the visual representation associated with the shift in system or application modes. This is also referred to as temporal consistency or stability (Mahemoff and Johnston, 1998b) and should be considered visually as well as mechanically. If the need to shift modes will prevent the user from actioning current items or to operate within a singular functional path then this should be clearly indicated by the visual representation at that time. This can be achieved through techniques such as setting of focus, establishing an obvious active foreground and inactive background areas and through clear delineation of the currently active area from the inactive components of the system.

5.6.5 Influencing Interface Systems

User expectation and therefore the design of interfaces for virtual and augmented reality environments must take influence from the interfaces of computer games and those presented in movies. There have been many films which have specifically depicted virtual reality in a populist sense. These include The Lawnmower Man (Leonard and Everett, 1992), The Matrix (Wachowski and Wachowski, 1999) and Tron (Lisberger and MacBird, 1982) to name a few. Most movies portray a world well beyond our current capability and create the impression in the public mind that immersive VR will be photo-real, feature fully reactive physics engines and will be a near seamless experience cognitively. Some films such as The Minority Report (Dick and Frank, 2002) and Final Fantasy: The Spirits Within (Sakaguchi and Reinert, 2001), portray possible augmented environments and provide beautiful visual reference though little reality in expectation setting of what is currently possible.

Video game use is widespread, with approximately two thirds of males in the US claiming to be regular users (Entertainment Software Association (ESA), 2005) and similar number in Australian markets. Games have introduced complex interfaces presented in multiple modes which build upon the expectation of users for virtual and augmented environments. Games with rich immersive environments such as Half Life 2 (Valve Software, 2005) have built a base expectation for many users for the visual and physical quality which can be expected. While these types of interfaces may be simple compared to those of real world industrial or scientific needs, they can provide clues to user preferences or a paradigm for visual representation which will assist the user in familiarisation with the VR or AR.

Aside from these more novel presentations of user interfaces, users and designers will be influenced by the other systems they currently use on a regular basis, that is, desktop computers and mobile phones or media players. These devices are so common in developed countries that users will interact with them on a near to or daily basis. This learnt environment will have a heavy influence on the expectations of users as they will attempt to apply their understanding of these device interfaces when using an AR or VR interface.

5.7 Summary

Within this section a range of information has been presented to build a base knowledge and context for the development of 3D interfaces. We have seen that navigation is based on two major processes; cognitive mapping and spatial ability. The common tasks types are exploration, search and manoeuvring. We have seen that navigation techniques can be divided into two core components, travel and wayfinding each having subsets of specific metaphoric techniques which relate to the way in which we interact with our physical and cognitive models of the world.

Two major components of interaction exist; selection and manipulation techniques. Variations in the types of current tools and concepts for these interaction types have been explored including the use of multimodal techniques and the use of both physical and virtual tools.

A summary of the nature of system control tools and interface components has been investigated showing that the range of options are broad when designing a specific application which has specific purpose. These tools can include the use of various types of input and display devices, all of which are suitable for differing purposes.

The notion of pattern languages has been introduced as a method for obtaining consistency across executions and assessments. This has been shown as useful in the realms of architecture, programming and usability and has been applied to the structure of tools, tasks and users to build interfaces variations based on consistent structure and principle. We have discussed the way in which these interfaces can be represented and the tools used for the representation. An understanding has been drawn of the nature of influence of external interfaces and popular culture as environments which set expectations for possible user groups.

The following section of the document works through the development of an experiment where the topics explored above can be put to use and investigated further. Drawing on the information above, an experiment will be devised which will allow interfaces to be developed from consistent structure and principles allowing the visual variations to be accessed.

6 Experimental Approach

6.1 Planning

This section contains the definitions of patterns for the overall project, the users, the tasks and the individual designs as well as an extended development of the actual designs, the tasks to be performed and the environment within which the experiment will take place.

According to Mahemoff (Mahemoff and Johnston, 1998a), the key attributes of a pattern are:

- Name: A name to identify the pattern.
- Context: The situation(s) where the pattern is relevant.
- Forces: The forces present which may constrain or suggest alternative solutions. When these forces are in tension with one another, the problem is harder to solve and a compromise may be necessary.
- Solution: A solution which resolves, as far as possible, the various forces.

These definitions are derived from the original works on the nature of patterns and pattern language by Alexander et al (Alexander et al., 1977) where this system of pattern usage was developed with respect to building and town planning.

The series of patterns to be defined are:

- Patterns of tasks
- Patterns of users
- Patterns of tools

From these pattern definitions, a series of interface designs can be constructed with respect to a consistent set of criteria. This will allow for a more consistent view of the data collated and ensure that the same underlying principles are applied in each instance.

6.1.1 Timeline

An approximate timeline for the completion of the experiment is provided below. Times are listed in approximate labour days, i.e. 8 hours work, rather than calendar days.

Interface Variation Testing	**132 days**
System Design and Build	**86 days**
Completing interface designs x 6	24 days
Interface variation build times within VR/AR	60 days
Setting up the testing environment - location	2 days
Measuring and testing framework	**8 days**
Test design	2 days
Integration into the actual system for the trials	6 days
User Group requirements	**16 days**
Sourcing appropriate subjects	4 days
Assessing each subject x 6 tests each	12 days
Analysis	**22 days**
Data collation and analysis	14 days
Reporting	4 days
Assessment revision where appropriate	4 days

Table 1. Development timeline example.

This timeline makes the assumption that the participants (not the users) in the experiment are experienced in the design and development of interfaces, applications and the chosen system environment. If the participants are not versed in the development of this type of project then the initial two stages would be expected to take longer by the degree of skills acquisition required. The timeline could be reduced if the development of the system was based on an existing platform which allowed for rapid development and deployment based on existing code libraries or development engines.

The team expected to complete the experiment would consist of the Research Project Lead, Interface Designer, Programmer, Research Assistant and the appropriate users as required. Some of the above roles could be shared by a single person.

6.1.2 Objectives

The key objective of the study is to assess the nature of influence of differing visual design treatments of a common interface structure as it relates to the users perception of comfort and ability to perform a specific set of tasks.

Secondary objectives

- To assess the transfer of understanding of a subject when presented with varying designs of the same interface.
- To use design patterns to provide a framework for development of visual design and usability structures.
- To provide insight into preferred visual mechanics of interface elements of users within VR and AR environments.
- To provide information from which some guidelines can be drawn for interface design in three dimensional environments.

6.1.3 Criteria of Assessment

The designs will be assessed through both the observations (qualitative) of the user after the completion of the task set and the measured responses, durations and accuracies of the task performance (quantitative).

This assessment will specifically measure:

- Time taken to become familiar with new design
- Time taken to perform tasks
- Accuracy of the task completed
- The user will complete a questionnaire after each task set completion. This will include questions relating, but not limited, to:
    - Visual appeal
    - Ease of understanding

- Ease of recognition
- Influence of past learning
- Comprehension
- Comfort
- Perception of measured assessment items

6.1.4 Testing Task Definition

Taking lead from Mahemoff and Johnson(Mahemoff and Johnston, 1998b), the test tasks will consider the following:

- Task Efficiency: minimise effort to perform tasks.
- Reuse: reuse existing effort and knowledge.
- User-Computer Communication: appropriately representing changes to the system.
- Robustness: minimise misperforming tasks and facilitate recovery from errors.
- Flexibility: allowing for multiple ways to perform tasks and by enabling user customisations of the system.

The test tasks will be based around the performance of several core functional user processes, interaction with the system, the application and specific elements with the application. The specific tests are defined further below and individual tasks are defined within the pattern structures.

For the initial study the experiment will use a fixed set of tasks. This will include interaction with the system, the application and specific elements. This restriction is put in place to ensure that the results drawn reflect the base hypothesis, that is the ability to specifically assess the preference through consistent interface structure and principles. Once this has been established then further study can be completed which will allow for testing a range of variations of task and interface designs which would highlight the preferences for specific representations for specific tasks.

6.1.5 Interface Design Variations

The range of interfaces that will be developed will be bound by the common framework as laid out in the pattern structure. The visual components of the structure will not be defined in this document beyond specifying the necessary elements to appear on any screen, menu or tool. This will allow for consistent interface elements throughout a sweep of visual design treatments. The variations should combine a range of subtle variations based on similar layouts through to highly varied interpretations of the patterns defined. This range of variation will allow for the subjects to show familiarity, translation and learning where appropriate.

The use of auditory, haptic and visual feedback should be consistent though the treatment may vary to suit the particular interface design. This will reduce the level of variation in the expected outcome from each task set.

When specifying the alternate interfaces reference can be drawn from Mynatt's graphic model (Mynatt, 1995) so that particular elements can be altered. Looking at this breakdown would give a direction for variation such as:

Element	**Variable Feature**	**Possible Variant**
Objects – that is the base elements	The nature of the objects, the kind of tools and core elements of their representation i.e. menus, pointers, sliders, polygons, plane or spherical and so on.	level of visual abstraction, animated verses static, 2D or 3D, different contextual settings
Visual attributes of the objects	The visual attributes which are common to the differing types of objects such as size, orientation, position and colour and so on	Colour, transparency, size, rotation, dimensionality, feedback allowances, texture
Affordances of the objects	The visual cues which imply functionality	feedback indicators, icons, placement, clustering, texture, dimensionality, textual association
Relationships between objects	The physical relationships such as proximity and the functional action associations derived through interaction	Placement, clustering, texture, responsiveness, focus
Names of objects	Not applicable	

Table 2. Interface design variations.

6.2 Analysis

The experiment will use a combination of assessments. This is achieved through both the qualitative testing of the user and the quantitative, measured responses, durations and accuracies of the task performance.

The user's reaction to the differing test interfaces will be assessed in a qualitative manner via the use of Likert scales in a questionnaire format. This was determined as the most appropriate method of analysis due to the simplicity for the user and the consistency of results for later analysis. This method as commonly employed in similar testing scenarios (Ergen, 1996, Lindeman et al., 1999b) and provides feedback which can easily be collated for large sample groups. Each user will receive the questionnaire as shown in Appendix D after every execution of the test.

The quantitative data capturing the user's behaviour will be collected within the experimental interface by the software. This will be built so as to record specific actions and parameters within the tasks at run time. The analysis of the quantitative data will be completed using a combination of t-tests and analysis of covariance (ANCOVA) as described below.

The combination of analysis as described in this section will allow the testing of the hypothesis. That is, the variation in visual representation will be able to be more specifically assessed due to the consistency of the interface's principle and structure.

6.2.1 Independent Variables

The set of independent variables used within this experiment can be defined as:

- Interface designs options (6 options)

The extended set of independent variables, which could be assessed in an extended study, are:

- Task allocation set (6 options)
- Interface characteristics, which will each need to be categorized
- User level (3 options)

The interface design options are the six alternate interface designs to be tested. To allow these to be identified they will be given unique identifiers as shown in the table below.

In order to simplify the task allocation variations, a set order can be used. At this stage the only test result required which relates to the hypothesis is whether the interface variations affect the users ability to complete tasks overall, and that this can be clearly identified through this process. For the initial rounds of testing the task set will be reduced to one set combination, this reduces the compounding factors will may be of interest at later stages.

For example – open, access tool, selection, manipulate (rotate) and save/close. This leaves room for future testing where three variations for the type of manipulation executed by the user are available. This also reduces the variations driven by the limitless variation of task combination available through more complicated task combinations.

The design options could be further broken down into a group of variables which could each then be categorised i.e. interface 1 = set colour, size, dimensionality, texture, clustering, placement etc. These elements would then also need to be given categorized identifiers so the variations can be differentiated.

User levels have been determined to fall within three categories, advanced, intermediate and novice as determined from the analysis of the user selection test shown in Appendix C. These should be recorded against each user and could be analysed separately revealing further information though is not necessary as this is not critical to the case of the hypothesis presented here.

6.2.2 Dependent Variables

The set of dependent variables used within this experiment can be defined as:

- Time taken to perform tasks
- Accuracy of the task completed per design.

An extended set of dependent variables, which could be assessed in an extended study or through alternate analysis, are:

- Time taken for alternate or individual tasks
- Correlation between, interface designs, error rate and time taken
- Accuracy of alternate or individual tasks
- The effects of user type against the above

Time taken to become familiar with new design will be incorporated into the task time recording as the measurement is the overall task rather than the individual subtasks.

Interface Variations	**Unique Identifier**	**Task allocation sets**	**Unique Identifier**
Interface 1	INT1	Open, select, rotate, scale, colour, close	TS1
Interface 2	INT2	Open, select, rotate, colour, scale, close (Not used in the initial study)	TS2
Interface 3	INT3	Open, select, scale, colour, rotate, close (Not used in the initial study)	TS3
Interface 4	INT4	Open, select, scale, rotate, colour, close (Not used in the initial study)	TS4
Interface 5	INT5	Open, select, colour, rotate, scale, close (Not used in the initial study)	TS5
Interface 6	INT6	Open, select, colour, scale, rotate, close (Not used in the initial study)	TS6

Table 3. Table of Independent Variables and categorisations.

Each combination of interface and task set should be tested with at least one subject of each user type. With three user types, six interfaces and six task sets the experiment would require a minimum of 108 combinations. This experiment should use the range of tasks and interfaces in varied order so as not to distort the findings through progressive learning applied to any one interface or series of interfaces.

For the initial study, which will contain only one task set, each user will be delivered one of the six interfaces in random order. This order will be recorded and is used as the covariate variable in the ANCOVA analysis.

The initial study will require a sample size of 18 at a minimum. This accommodates the base requirement of one user of each user type experiencing each of the interface variations.

6.2.3 Method of Recording Data

The two key components of the data to be maintained are the recording of the time taken to perform the overall task set and the number of errors that occur.

6.2.3.1 Time based measurements

The application will be developed so that it records the times taken to complete each interaction task set. To allow the start and stop of this timer, a control will need to be included for the experiment supervisor to initiate the timer and stop it at the ends of each task set. To further refine this measure and to gain deeper understanding, the time taken to perform each subtask could also be recorded. This can be achieved by allowing the system to have consistent, design neutral screens which allow the start and stop of each process and therefore the time record.

For instance, a wait screen is displayed when the user first initiates the system. A command to open a particular file is given. The user acknowledges the command and when the wait screen is removed, the timer begins. Once the subtask of opening the file is complete the wait screen reappears and the user can now be informed of their next subtask. This process may interfere however with the user’s sense of continuity for the current interface variation and has therefore been left out of this experiment. It is suggested that this method be used as a second stage refinement of the initial findings when altering the elements of a specific interface design, the style of which could be included in the wait screen to maintain continuity.

6.2.3.2 Recording of Task Accuracy

The level of accuracy can be tracked by recording the number of attempted interactions the user makes in error while completing a task. While this could be achieved through an automatically generated system within the application, it could be compiled by the experiment supervisor who will be observing the experiment. This variable will represent therefore the number of errors which occur during the task completion.

According to Hix et al, (Hix et al., 1999) this type of recording is typically referred to as the record of critical incidents.

“Qualitative data are typically in the form of critical incidents. A critical incident occurs while a user is performing task scenarios, and is an event that has a significant effect, either positive or negative, on user task performance or user satisfaction with the interface.”

This can be achieved as described above through the experiment supervisor’s manual recording or through more sophisticated means. Eye-tracking has previously been used in many interface development environments and has been introduced to the virtual environment through experiments such as Gabbard et al’s CAVE experiments (Gabbard et al., 1999). This type of analysis is often coupled with recorded path and event tracking analysis which allows the experimenter to replay the actions taken by a user after the fact for further analysis. This, when used with the eye tracking tools, allows the experimenter to observe the user’s eye movements in conjunction with the actual process they follow to complete the task in a format which can easily be replayed.

6.2.3.3 Post Test Evaluation Survey

The user will complete a questionnaire after each task set completion. Likert tests for interface preferences have been used many times for similar studies (Ergen, 1996, Grönegress et al., 2001, Lindeman et al., 2001) and provide a method for users which are intuitive, quick to complete and clear to answer.

This will include questions relating, but not limited too:

- Visual appeal
- Ease of understanding
- Ease of recognition
- Influence of past learning
- Comprehension
- Comfort
- Perception of measured assessment items

This testing is conducted using a set of questions structured using Likert scales for ease of use for the user and for the simplicity and consistency this brings to collation of data. This questionnaire can be found in Appendix D.

6.2.4 Method of Comparison

The experimenter can separately analyse the variations in time taken and accuracy against the alternate interfaces using ANCOVA. This is complete with the consideration of order of interface delivery as the covariate variable.

From this analysis the experimenter will be able to assess:

- The relationship of the time to complete relative to the interface design used.
- The relationship of the error rate relative to the interface design used.

Further analysis could be completed from this data using multivariate analysis of covariance (MANCOVA). This would allow for the analysis or the interface variations and their relationship to time taken and error rate in combination.

If the variation in time taken or the error rate is statistically significant different we can conclude that the interface design is the confounding factor.

It is possible that during this round of experimentation we are to also record the user type, all possible combinations of variation within the design (categorised into characteristics) and the variations in task set combinations. Through this a deeper analysis could be made of the correlation between individual visual elements and their affect on the user’s ability to complete tasks overall. While this is outside the scope of this study it does provide a basis for future work.

6.3 Design

The design section will cover the definition of the experiment design and the specific sub elements of each component. First we review the nature of the pattern structure as it relates to the experiment and provide examples of the core patterns to give context for the discussion. Further definition of the set of patterns is included in Appendix A, some of the information stated in this section is repeated in the appendix to maintain readability of the pattern collection. This is followed by defining the tests, which are the specific test cases and the variations plus how these will flow for each user type.

6.3.1 Pattern Definitions

In this section the patterns will be defined which will be used as the reference point for the construction of the various interfaces, the tasks, users and tools. Through a common set of definitions as described below the series of varied interface treatments will be developed, each one adhering to these patterns so as to provide some consistency in the process, behaviour and experience being observed.

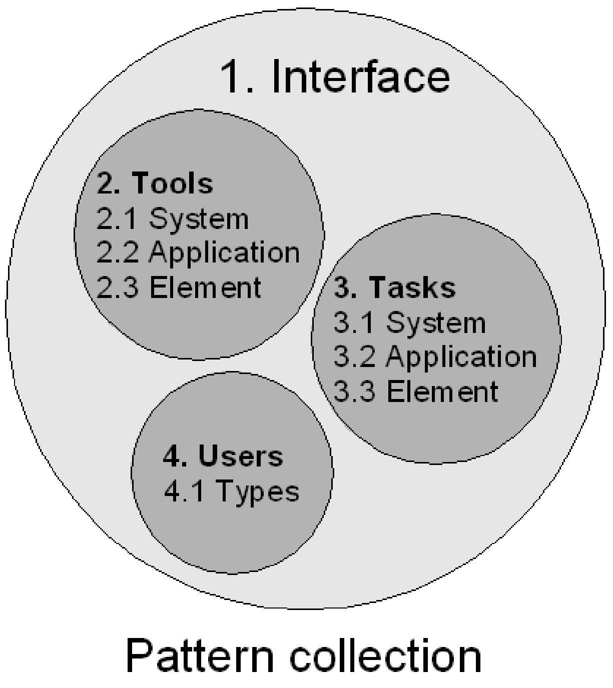

Figure 3. Structure of the patterns collection used within this document.

6.3.1.1 Creating the patterns

There are many templates or guidelines which form the structures which are generally considered to be labelled patterns or elements of a pattern language. These elements are all derived from the original patterns structures as presented by Alexander (Alexander et al., 1977). The original structure of these can be summarised into:

- providing the context – that is what are the conditions of the problem which exist
- system of forces – this is a declaration of constraints or influences
- the solution – a stated method which will address the problem with respect to the forces as shown

In the works by Alexander and following this the programmatic world with the works of the Gang of Four (Gamma et al., 1995) and so on there have been many differing renditions of the appropriate details to include with each pattern within a collection or language. It is common practice to include the names of the patterns to refer to both before and after the current pattern which further provides context and to provide clear variation if these exist (Appleton, 2000, Beck and Cunningham, 1987).

A pattern structure was required to achieve the underlying design structure for the experiment used in this study and as a reference several templates or guidelines for pattern definitions including:

- Alexander(Alexander, 2001)
- Mahemoff and Johnston (Mahemoff and Johnston, 1998b)
- Gamma (Gamma et al., 1995)
- The Gang of Four's Template (Frye et al., 2005) .
- Doug Lea's html template (Frye et al., 2005).
- AG's HTML template (Frye et al., 2005).

From this we can build the base structure for the set of patterns for this experiment and using the common structure for programming and design patterns which will be covered as such:

Name: A clear, unique identifier

Context: Simple definition of the conditions

Consider these patterns first: Named patterns to provide context

Problem: Base problem this pattern addresses

Forces: The known constraints

Solution: The general solution.

Consider next: Patterns to continue the context

The patterns for elements and tasks will have the extended framework referencing the usability pattern structures similar to Hussey (Hussey et al., 2001), including.

Task Efficiency: minimise their effort to perform their tasks.

Reuse: Ensure that users can reuse existing effort and knowledge.

User-Computer Communication: Facilitate collaboration between humans and computers.

Robustness: Minimise misperforming tasks and facilitate recovery from errors when they do occur.

Flexibility: Account for users of different backgrounds and.

The two sets of structure have now been combined within this document to define a set of patterns from which the experimental can be executed.

The patterns described in Table 4 are an entry point to the development of a further set of patterns for interface design in VR and AR environments, however the below is limited to the scope of need of this experiment rather than the global need of the field.

Overview		
		CoreInterface
Tools Patterns		
	Global pattern	Tools
	System	
		ToolsOpenFile
		ToolsCloseFile
	Application	
		ToolsApplicationMenu
		ToolsApplicationSelectObject
	Element	
		ToolsFocusObject
		ToolsSelectObject
		ToolsRotateObject
		ToolsColourObject
		ToolsScaleObject
Tasks Patterns		
	Global pattern	Tasks
	System	
		TasksOpenFile
		TasksCloseFile
	Application	
		TasksApplicationAccessMenu
		TasksApplicationSelectObject
	Element	
		TasksSelectObject
		TasksRotateObject
		TasksColourObject
		TasksScaleObject
Users Patterns		
	Global pattern	Users
	User Types	
		UsersAdvanced
		UsersIntermediate
		UserNovice

Table 4. Structure of the patterns collection used within this document.

6.3.2 The Central Patterns

The experiment is designed around three central pattern groups which are a subset of the central overarching core interface; the Tools, the Tasks and the Users.

6.3.2.1 The CoreInterface Pattern

The `CoreInterface` pattern describes in broad term the requirements of the system and its subparts overall. This is the general binding description of the system itself and presents the relationship between the sub parts of Tools, Tasks and Users. This pattern also presents the base structure for the other pattern definitions and shows for the first time how the differing pattern elements are addressed.

The key points found when assessing the `CoreInterface` pattern are:

- The interfaces should allow Users to complete Tasks using a collection of Tools
- Tools and Tasks should be developed to allow for developed and applied efficiencies
- Users should be able to learn and translate knowledge within the system
- The system should provide feedback, be robust and comprehensible.

The three high level patterns which then describe the sub parts of the experiment are presented here and again in the Appendix A. These are included to provide an understanding of the approach taken throughout the remainder of this document and ti give context to the reader.

Name: `CoreInterface`

Context: This is the framework through which the user will interact with the system.

Consider these patterns first: This is a top level pattern

Problem: The `Users` require the ability to manipulate, interact with and control data in various representations (`Tasks`) via a collection of `Tools` which is flexible, comprehensible, robust and efficient.

***Forces*:** The `Users` will have varying levels of ability and understanding of the context of the system and the `Tasks` presented. The `Tools` will need to be built with enough flexibility to be able to be used by this `Users` range of skills and be consistent with other existing and yet to be developed `Tools`.

***Solution*:** Provide a means through which the `Users` have access to `Tools` which allow the performance of `Tasks` through a representation which is consistent, comprehensible, stable and comfortable. This will be achieved through the application of the available guidelines and instruction as provided here and in the supporting texts.

***Consider next*:** `Tools, Tasks, Users`

***Task Efficiency*:** At all times the interface should seek to minimise the `Users` efforts to perform their `Tasks`.

***Reuse*:** Ensure that `Users` can reuse existing effort and knowledge through the use of consistent and predictable representations.

***User-Computer Communication*:** Wherever possible, the use of appropriate feedback should be applied. This may be delivered through either singular or complimentary senses, i.e. haptic, visual aural.

***Robustness*:** Minimise misperforming of `Tasks` and facilitate recovery from errors when they do occur through the clear representation of function and modelessness of operation. `Users` should also be forewarned of any action which may cause data loss or has no return path and where this does exist it should be clearly accessible.

***Flexibility*:** Account for `Users` of different backgrounds and capabilities.

6.3.2.2 Patterns of Tools

This is the limited collection of patterns describing the expected set of tools which will be available or necessary to under take the experiment. These are presented as high level explanations of the mechanical function required to action the specific task to which the tool relates.

The general tool pattern defines the overarching structure for all tools and defines the functions and expectations which are then inherited by all Tools patterns.

***Name*:** `Tools`

***Context*:** The interface structure and devices required to perform specific `Tasks` are considered to be the tools. This set of patterns is specific to accomplishing the `Tasks` within this experiment and are limited to that end.

***Consider these patterns first*:** `CoreInterface, Users, Tasks.`

***Problem*:** The `Users` need a method for actioning the desired `Tasks`. The `Tools` provide a means of human computer interaction.

***Forces*:** The experiment itself takes place within a controlled and predetermined environment which will bring with it a set of constraints in terms of input devices, displays, feedback and implied understanding. It is attempted throughout this section to be non platform specific but the assumption must be made that the `Users` will have visual feedback with optional haptic and aural feedback plus a visual display of some type which will be either a VR or AR display device. The `Users` will also have the ability to interact via a selection device such as a mouse, tablet or similar allowing the `Users` to make selections and interactions with the presented `Tools` through a physical process.

***Solution*:** The `Tools` must indicate the process through which `Users` can perform `Tasks` in a way which is clear to transfer into the user interface design. These `Tools` should follow generally accepted usability practices or state where they differ if this is a novel variation.

The `Tools` will be defined through three components, the system, application and elements. These represent the three separable areas which will normally be treated in a similar fashion throughout most environment designs. That is the system has its own paradigms which are broader than the application currently in use, the application itself will have a range of specific functional and visual traits and the interface elements themselves will have a combination of novel, application and system influenced representations.

***Consider next*:** `Users, Tasks, ToolsOpenFile, ToolsCloseFile`

***Task Efficiency*:** These must allow `Users` to complete `Tasks` within their current workspace wherever available and with the minimum amount of effort required.

***Reuse*:** Where possible existing tool paradigms from existing systems or the real world should be applied so that `Users` can reuse existing effort and knowledge. This should not be adhered to if the resulting `Tool` is more nostalgic than functional.

***User-Computer Communication*:** Facilitate collaboration between humans and computers.

***Robustness*:** All `Tools` should provide the necessary internal constraints and appropriate feedback for errors or options.

***Flexibility*:** Where possible the `Tools` should have multiple activation or interaction styles which cater for the differing levels of `Users` ability. This is often presented in the form of shortcuts for `UsersAdvanced` and Wizards for `UsersNovice` who may be less skilled.

6.3.2.3 Patterns of Tasks

This is the collection of patterns describing the framework of actual asks which will be demanded of the users who undertake the experiment. These are presented with examples of request structure and fall into the general categories of system, application and element level tasks.

Name: `Tasks`

Context: `Users` are presented a range of scenarios in which they need to achieve some end goal, that is, complete a specific task. This is accomplished by utilising the available and appropriate `Tools`.

Consider these patterns first: `CoreInterface, Tools, Users`

Problem: The need to achieve the specific end goal, in this experiment this is multilayered in that there are a series of `Tasks` for `Users` to complete each with their own goal and the overarching task which will allow for the adequate testing of the interface which has been developed.

Forces: The `Users` will need to use only the `Tools` provided to achieve the goals stated. The `Users` will have varied levels of ability and prior knowledge of how to complete each of the `Tasks`.

Solution: Provide clear definition of the end goal and any surrounding parameters which need to be considered to execute the task. Where `Tasks` are multi-step present them in a context which indicates this unless the discovery of the process is part of the task itself.

Consider next: `Tools, Users, TasksOpenFile`

Task Efficiency: Where possible the `Tasks` should be presented in an order which promotes efficiency rather than rework.

Reuse: Ensure that `Users` can reuse existing effort and knowledge by providing similar clues to the end goals and by allowing the use of similar `Tools` to complete multiple `Tasks`.

***User-Computer Communication*: **Provide clear indication that the `Tasks` have been completed correctly or not by stating clear end goals.

***Robustness*:** If `Tasks` are attempted and not completed then there must be the facility to attempt them again.

***Flexibility*:** Where multiple options for completion exist do not be so descriptive in the `Tasks` presentation that the `Users` are forced to follow a step by step process to execute the `Tasks`.

6.3.2.4 Patterns of Users

This is the collection of patterns describing the expected user group and the types of users to under take the experiment. These are presented as three differing user types, advanced, intermediate and novice.

***Name*:** `Users`

***Context*:** This is the collection of people who are to actively use the system. They have varied familiarity with computers and may have been exposed to a range of applications, interfaces, input devices and task types.

***Consider these patterns first*:** `CoreInterface, Tools, Tasks`.

***Problem*:** A collection of users are required to provide feedback from range of viewpoints which may or may not have been exposed to similar experiments or interface developments. This allows the experiential feedback to be provided from multiple perspectives and without the bias brought to the interface by specific past experience.

***Forces*:** The user types are ranged into novice, intermediate and advanced to ensure there are a range of user types. It is best for testing to have an even spread of these types.

***Solution*:** Subjects should come from a range of backgrounds, some with minimal exposure, some who use computers regularly and some using computing professionally, as a hobbyist or computer game enthusiasts and with a broad exposure to various system paradigms. Predisposition to any particular environment across the collection of `Users` could reduce the accuracy of the results.

***Consider next*:** `UsersAdvanced, UsersIntermediate, UsersNovice`

6.4 Testing the Interfaces

6.4.1 Test Variations

To provide a generic set of tests which will apply to all the interfaces available, independent of their design, the following series of tests will be conducted. It is intentioned that these tests can be applied through the use of the Tasks pattens allowing for consistency of implementation.

A sample test is presented in Appendix B – Sample Test.

All tests should follow the basic structure as outlined below, that is they will complete a series of tasks described below and using the task patterns.

System Interaction Test

This will include two tasks:

- Opening a previously created scenario, file or dataset.
- Saving and closing the completed scenario, file or dataset.

Application Interaction Test

This will include two tasks:

- Accessing a variety of tools for manipulation
- Accessing a variety of tools for selection
- Accessing commands for opening and saving the scenario, file or dataset.

Application Element Interaction Test

This will include, at minimum, four tasks:

- Selecting an object and completing a rotation manipulation.
- Selecting an object and completing a colour manipulation.

- Selecting an object and completing a dimensional manipulation.
- Selecting an object and completing a shape manipulation.

6.4.2 Situational / Environmental Restriction and Structure

The nature of the tests as outlined in this document are designed for a system with applications which are viewed generally in the first person and makes use of a visual display and a physical input device. While it is possible to use any type of input device to perform the tasks as expected this would be dependant on the actual test devices available and the environment in which the system was to be used.

The proposed tests would take place in a closed environment, such as a specific lab cubicle, where the user was shielded from external distractions and influences. The more varied nature of a more open environment would lead to variation in the results and inconsistency in the user's responses. While there is certainly benefits for similar testing to be carried out in environments which are not controlled, such as streetscapes, operating theatres or battlefields this would be best completed after a set of base line studies had been successfully completed.

It is understood that the set of tests, which may occur at different times of the day and indeed in different, yet similar, physical locations. With this in mind however, the actual nature of the input devices and display combinations should be kept as consistent as possible throughout the experiment to ensure consistent results.

6.4.3 Mechanical Design and Process Description

A specific description regarding input and display devices for the experiment have deliberately been excluded from this document. This has been done due to the fact that the equipment, devices and associated experiment varies greatly in this field and is under constant revision. As the author has no direct access to an experimental lab, no specific equipment has been cited. This allows the experiment basis to be adapted for any particular environment and the currently available equipment.

This also removes the issue that the technology at hand is continually changing, many of the higher end devices available at the commencement of this study have since become superseded. An example of this can be seen in 3D displays, in the last five years many of the major manufacturers have invested in and either have or near to released stereoscopic displays for desktop “Fish Tank VR” use.

The required steps to take before executing the test are presented below. This will need to be revisited to take into account and environmental and mechanical specific requirements as necessary.

Pre-test tasks:

1. Define the specific testing equipment; input and display devices.
2. Define the application environment; AR or VR and the application type and purpose.
3. Define the application development environment and limitations; programming language and tools, access to expert knowledge and code libraries.
4. Define the expected variations in visual representation treatment with respect to the pattern definitions and the system limitations. Assign each an identifier.
5. Screen all available users and collate into specific user types.
6. Define the selection of tests verses the selection of users. Every user will be delivered every interface variation and task set at least once, that is 36 separate participations per subject. A more accurate result would be determined if subjects we tested repeatedly on each interface and task combination. Order of assignment should be randomised to reduce effects of learnt responses.
7. Define the overall order of tests and prepare an error record log. A simple table indicating a single mark per error for each test per user is sufficient.
8. Organise testing location requirements and ensure availability and access for the duration of testing dates. Ensure there are some contingency time for overflow or re-testing if required. If the location cannot be left setup between testing sessions ensure that a procedure is in place for the assembly and disassembly of the experimental environment to improve consistency. Ensure there is a suitable position for observing the subject and recording errors.

9. Define a brief induction for test subjects with the specific devices and application in mind. This should cover; a "how to" for the input devices and what to expect from the display types, especially if they are novel.

10. Define a walkthrough explanation for the test subjects containing information of the session timeline. Including the flow of the experiment from their perspective, the induction, the test, the post test survey and the next steps for repeating the test with new interfaces.

11. Ensure enough copies of the post-test survey are available for the session.

6.4.4 Flow Process for Test Iterations

The procedure for each round of user testing should remain consistent and follow the basic steps as outlined below. This assumes that the subjects have been preselected, categorised and assigned the list of tests to be completed.

For each test session:

1. Welcome the subject and introduce them to the test environment.

2. Use the session walkthrough to inform the subject of the requirements for the session and let them know what they can expect.

3. Run through the induction material to familiarise the subject with the test environment and equipment.

4. Provide an opportunity for the subject to ask questions or to clarify what has been discussed, be careful not to provide more than the predefined information however as this may influence responses.

5. Initiate the test environment; place the subject within the environment and initiate the correct interface variation.

6. Explain the required task set to the user.

7. Allow the subject to initiate the test, starting the timer.

8. Observer the subjects actions and record on the test record sheet any navigation and interaction errors.

9. On completion of the task set, ensure that the subject has finalised the task and therefore stopped the timer.

10. Assist the subject with disengaging from the devices in use and provide the post test survey.

11. Ensure the survey has been completed and returned commencing the next test.

12. If the subject has completed the final test for the session ensure that all surveys and tests for this session have been recorded correctly. If so, thank the subject and arrange the next session if required. If not the subject should resit the test which has been incorrectly recorded before ending this session.

13. Repeat this process as required to collate all the required data.

6.4.5 Definition of User Types for Testing and How These are Determined.

The differing user types sourced for the tests are explained in the User patterns in this document both above in outline and below in detail. These users will be volunteers ranging from the advanced, experienced computer users through to novices with limited exposure to computing especially in terms in 3D environments.

This range should allow for better assessment across the possible user range and will help to stabilise the result set which may be distorted through the testing of a single particular user type. Each of the user types will be given the same level of instruction and assistance, which will be aimed at the intermediate user, therefore forcing the novice to self discover some elements and the advanced user to be provided with more information than may be necessary as they will naturally apply their previous learning.

The actual test for screening users and determining their level of competence is provided in Appendix C. This questionnaire format is derived from a similar test scenario used by Ergen when determining characteristics of users for web browsing experiments. (Ergen, 1996)

7 Summary and Conclusion

7.1 Summary

Through a review of the literature and the proposal of an experimental framework, this study has argued for the development and assessment of virtual interfaces using patterns. It has shown that through the provision of consistent structure visual representations can be derived and their merits assessed.

Taking lead from the body of research by Sutherland, Bowman and Gabbard this study embraces the development of common reference point for interface and interaction development. An approach to 3D widget design has begun (Döllner and Hinrichs, 1998), though mostly from a mechanical perspective and some (Lindeman et al., 2001) have indicated links between ease of use and props. Several studies have been reviewed which provide usability characteristics (Gabbard, 1997) and build taxonomies of interaction tasks (Bowman, 1999). Through the literature review, little study has been found in the area of visual representation within virtual and augmented environments. Some studies (Gabbard, 1997, Lindeman et al., 2001, Wloka, 1995, Wingrave, 2001) attempting to pre-empt the types of interaction techniques exist, yet this work on visual structure has been somewhat secondary, this is perhaps appropriate as the foundations are required first.

It has been argued that direct imitation of real world interactions has been found not to be the most efficient process (Bowman, 1999). Many 2D interaction tools have been translated, though these need to be revisited due to the multimodal opportunities and visual context of interactions available in 3D immersive environments.

With this new paradigm of computing, especially coupled with wearable computing, the locations and environments in which devices are used are expanding. What were once non-digitally driven environments now can be and the arena for such types of computing will continue to expand. As such the representation of virtual elements in AR will greatly affect usage as they are consistently compared to the real world around them. The difference may enhance or deplete interaction depending on its purpose. For instance information bubbles above real world elements would be clearly virtual elements, whereas integrated virtual tools such as signposts providing path cues may be better when visually integrated so not to distract the viewer when not required.

This document is limited to investigating the visual design aspects of the three dimensional user interface elements. Therefore, before it is possible to begin investigating the many variations of interface representation a framework for the assessment must be established so that the differing representations can be assessed in a similar, replicable fashion. While this study has been restricted to a theoretical rather than practical realm, the experiment proposed is ready for execution and forms the first step of future research in this area.

The hypothesis presented here is, that through pattern definitions, 3D interfaces can be consistent in principle and structure and that through this consistency the effects of variation in visual representation on the user's ability to successfully perform set tasks can be more specifically assessed. To validly assess this, it is necessary to provide context, that is to explore the environment, the devices and the techniques which are the influencing factors, which contribute to the interface itself, and its use.

The literature review has provided this exploration and has defined context around six areas of enquiry, navigation, interaction, key interface elements, their use, representation and application.

It has been shown that navigation has two essential processes, cognitive mapping and spatial ability. Cognitive mapping, the understanding of the relationship of elements within the system, is improved through the use of visual and physical cues, which include paths, edges, landmarks, nodes and districts (Darken and Sibert, 1993). Spatial ability, the sensing and understanding of the space and the users location within it, is usually broken down into three areas, orientation, visualisation and relations (Ben Hajji and Dybner, 1999). These provide methods for both location and orientation allowing the user a sense of place and hence assisting in the movement from one location to the next.

The interaction techniques used to perform these processes are clustered under two core components, travel (motor) and wayfinding (cognitive). The user performs the actions of exploration, search and manoeuvring to complete these processes. It is through these actions that the user is able to traverse the system or environment. These can be facilitated through varied means some of which have been explored above.

Five common metaphors for travel interaction can be summarised as (Bowman et al., 2001) physical movement, manual viewpoint manipulation, steering (most common), target based and route planning. Wayfinding or the cognitive process of defining a path through an environment is the use of a user’s spatial ability to define and comprehend their environment(Barlow, 1999). This is aided by the use of cues and is a user centred and environment controlled process.

Interaction within VE and AR is generally based around the methods associated with selection and manipulation. While there are many techniques for selection available most fall within two main categorisations, ray-based techniques and reaching techniques, with a third, multimodal technique which is also being explored (Wingrave, 2001).

The above has shown that manipulation is the action of touching with the hands or mechanically means, with the intent to relay change. Common approaches include simple repositioning, scaling, rotating and generally altering the properties of the object in question. Elements of the interface which allow manipulation, tend to convey a sense of function in their visual representation and this will have varied levels of abstraction depending on the context of the function (Döllner and Hinrichs, 1998).

Manipulation should be represented as a separate process to the selection component of the interaction. It may be an extension or complimentary action and where possible make use of existing techniques. The use of 3D over 2D widgets and the introduction of props and at least simulated surfaces increased the user’s task completion accuracy (Lindeman et al., 2001). Of the many types of interaction tools possible, several have been highlighted which are of special interest or form common use. These include the TULIP system based around the flexibility offered by pinch gloves (Bowman and Wingrave, 2001). While not always practically applicable, the use of pen and tablet props has been seen to improve accuracy and comfort within VR manipulation tasks (Bowman, 1999, Serra et al., 1999, Lindeman et al., 2001). It is through the use of a tablet device that 2D widget tools are available in a fashion users can understand quickly. The surface interaction provides passive haptic feedback without excessive equipment but does however rely on the quality of the visual registration for seamless integration.

The use of floating menus takes its lead from the 2D desktop interface paradigm and since similar in function these are familiar to most users. The major difference in the 3D environment is the location of origin for the menu. Without the bound elements of the screen area, in VE and some AR applications the menu is usually bound to a specific point of reference to the user’s viewpoint. While this can cause visual occlusion the one menu may contain a deep set of nested commands which collectively take up minimal space when not in use and the menu is retracted.

As stated above, system control is any task where the user applies a command that changes the system state or mode of interaction (Bowman et al., 2001) and ideally the integration of system control is modeless or seamless. There are four base groups of interaction types for system control, which are graphical menus, voice commands, gestural commands and tools.

Graphical menus are the most familiar form of controls and borrow heavily from the 2D interaction models. Voice commands, also available in 2D interaction are appealing in 3D environments because they are not coupled to physical device presence or representation. While these may be flexible, have multi-level functions and complex menus, which are accessed via speech, they are difficult to learn. Data gloves and similar input devices allow the use of gestural commands. To become effective these need to recognise both singular and sequences of hand postures, and adapt on the fly (Gabbard, 1997). Virtual tools can greatly increase the complexity of the interaction and can allow for complex object manipulations. The use of virtual tools is applicable for devices which mirror natural devices or perform “Magic” applications.

From a visual design perspective we can break the interface into key elements specific to function. The core areas of system access, navigation and task based activities command specific treatment.

System access in an environment should provide a point of interface consistency. This presents a challenge for interface development as the applications may be wildly varied. The consistency allows the user’s cognitive load for this element to be spread across the entire system rather than relearning for each application and allows users to more readily define the boundaries of the application.

System navigation can be split into the common elements for manoeuvring the system details and common task specific tools for inter-application processes. This may differ from the application level tools which may be restricted to viewpoint manipulation and travel functions.

The majority of users of applications with a GUI will recognise a three dimensional object with a text label as a button and expect it to function as one. With more degrees of freedom of interaction these common metaphors become more complex. Many common 2D interface elements for task-based interaction will translate directly into virtual space, such as buttons, sliders, menus and input dialogs. However, new variations will be required to accommodate the changes in input devices and displays.

As computing continues to progress and new interaction devices and requirements develop, the range of visual representation variations grow.

Of the many types of visual displays available some of the most common have been explored. Moving from the simple conventional CRT monitors used in conjunction with stereo glasses to the more compelling surround screen displays such as the CAVE (Cruz-Niera et al., 1993), autostereoscopic monitors and hemispherical dome displays. More suited to some specific task types are displays such as the workbenches and arm-mounted displays used for mainly fixed position or non travel based applications. For physically mobile or head tracked applications HMDs and virtual retinal displays are more suited though provide only single user experience per display, though these can operate within the one VE for a collective experience.

These displays coupled with haptic devices provide the sense of immersion usually sought after in virtual environments. Relying on either tactile cues like texture temperature or pressure, or kinesthetic cues such as position or movement, haptic displays can either reinforce or replace visual displays. It has been seen that even simple mechanisms such as basic vibrotactile feedback increases the sensation of interaction (Lindeman et al., 2002).

A sense of presence from a series of auditory cues (Warren, 2002, Barrass, 1998, Mynatt, 1995) has specific implications for users with visual impairments and has been found as the most efficient alarm signal (Lecuyer et al., 2002). In 2D interactions, auditory cues have been commonly accepted as increasing the engagement for the user.

A reference to assess usability characteristics was drawn up by Hussey (Hussey et al., 2001) and broke down the points of investigation into task efficiency, reuse, user-computer communication, robustness and flexibility. This notion has been coupled with the concepts drawn from the work of Mahemoff and Johnston (Mahemoff and Johnston, 1998b) which heralds the use of pattern languages for information transfer of usability. Together this has provided a framework for the development of not only consistent interface visual representation but its assessment as discussed in this document.

Each interface design, based on the pattern definitions for structure, is the specific visual treatment of the differing elements within the graphical model. Each object, its visual attributes, its affordances and its relationship to other objects can be varied based on the consistent structure.

The graphical representation may then be complimented with auditory treatment. Like the graphical representation these utilise abstracts, simulations and metaphors to allow for an appropriate level of communication of function or purpose. The use of earcons can be seen as similar to that of graphical icons and as such require that actionable areas are distinguishable from ambient elements.

The incorporation of physical tools into the interface design affords the benefits of passive haptic feedback and provides a placement area for elements of action. The visual appearance of the prop may vary dependent on situational or modal context such that the same device may visually have an interchangeable range of complex functions.

When beginning the development of visual representation in the 3D environment some lessons can be drawn from the 2D environment, such as the Apple interface guidelines (Apple Computer Inc., 2006) and similar. Visual feedback should be included system wide and can be improved with auditory or haptic feedback. Consistency, possibly the largest challenge for VR and AR, improves the user experience greatly. Applications should aim at a minimum to be consistent within their own bounds. Visual consistency aides the users perceived stability of the system. It is important to follow the generally held design principles for screen design to maintain aesthetic integrity, a challenge for all programmer led interface designs. Finally the notion of modelessness and therefore stability (Mahemoff and Johnston, 1998b) should be maintained visually as well as mechanically.

It should also be taken into consideration that user expectation for 3D interfaces is influenced by a range of experiences. The development of evermore sophisticated interfaces for computer games, mobile devices and portrayal in popular media all play a part in the context brought to the system by the user.

7.2 Conclusion

The key objective of the experiment presented in this document is for the assessment of various visual representations of interface and their affect on a user's task completion ability. The effects of the visual aspects are highlighted by applying consistency to the interface structure and mechanical principles through the use of patterns. It is through the use of patterns, the visual representation can be more readily assessed. These patterns are the basis of a larger set which would be developed through the experiment and ongoing development. Provided in Appendix A are patterns defining the key areas of which need attention for consistency in the experiment and those which are required for consideration throughout the design process. These are the tools, the tasks and the users.

Similar to a spoken language, a pattern language is evolutionary and is under consistent revision. As such the patterns provided here are a starting point and are not meant to be seen as a definitive guide. The enclosed patterns do however begin to inform the process which his required to build a set of interface designs ready for testing in the fashion outlined here.

Once constructed, these designs will be integrated into the end application and assessed by both the quantitative and qualitative methods presented. The combination of mechanical scoring and the Likert test based, opinion will provide a well rounded set of results as is often used in the field (Ergen, 1996, Lindeman et al., 1999b). It has been shown that the time taken to perform the tasks and the accuracy of the task completion can be measured as dependent outcomes of the visual design variations. It is expected that this testing will show results which favour one design over the other.

Once this has been seen, further analysis can be made based on the suggested extended set of variables in the experimental design. These include a range of variants of visual representations as they relate to the graphical model and the relationship to the specific task. The further study could provide information leading to the development of guidelines indicating the overall preference of one particular visual treatment over another for specific task types.

8 Appendix A – Pattern Definitions

This appendix includes the pattern structure breakdown as listed in the table above labelled, Table 4, Structure of the patterns collection.

8.1 CoreInterface

Name**:** `CoreInterface`

Context**:** This is the framework through which the user will interact with the system.

Consider these patterns first**:** This is a top level pattern

Problem**:** The `Users` require the ability to manipulate, interact with and control data in various representations (`Tasks`) via a collection of `Tools` which is flexible, comprehensible, robust and efficient.

Forces**:** The `Users` will have varying levels of ability and understanding of the context of the system and the `Tasks` presented. The `Tools` will need to be built with enough flexibility to be able to be used by this `Users` range of skills and be consistent with other existing and yet to be developed `Tools`.

Solution**:** Provide a means through which the `Users` have access to `Tools` which allow the performance of `Tasks` through a representation which is consistent, comprehensible, stable and comfortable. This will be achieved through the application of the available guidelines and instruction as provided here and in the supporting texts.

Consider next**:** `Tools, Tasks, Users`

Task Efficiency**:** At all times the interface should seek to minimise the `Users` efforts to perform their `Tasks`.

Reuse**:** Ensure that `Users` can reuse existing effort and knowledge through the use of consistent and predictable representations.

User-Computer Communication**:** Wherever possible, the use of appropriate feedback should be applied. This may be delivered through either singular or complimentary senses, i.e. haptic, visual aural.

Robustness**:** Minimise misperforming of `Tasks` and facilitate recovery from errors when they do occur through the clear representation of function and modelessness of operation. `Users` should also be forewarned of any action which may cause data loss or has no return path and where this does exist it should be clearly accessible.

Flexibility**:** Account for `Users` of different backgrounds and capabilities.

8.2 Tools

***Name*:** `Tools`

***Context*:** The interface structure and devices required to perform specific `Tasks` are considered to be the tools. This set of patterns is specific to accomplishing the `Tasks` within this experiment and are limited to that end.

***Consider these patterns first*:** `CoreInterface, Users, Tasks.`

***Problem*:** The `Users` need a method for actioning the desired `Tasks`. The `Tools` provide a means of human computer interaction.

***Forces*:** The experiment itself takes place within a controlled and predetermined environment which will bring with it a set of constraints in terms of input devices, displays, feedback and implied understanding. It is attempted throughout this section to be non platform specific but the assumption must be made that the `Users` will have visual feedback with optional haptic and aural feedback plus a visual display of some type which will be either a VR or AR display device. The `Users` will also have the ability to interact via a selection device such as a mouse, tablet or similar allowing the `Users` to make selections and interactions with the presented `Tools` through a physical process.

***Solution*:** The `Tools` must indicate the process through which `Users` can perform `Tasks` in a way which is clear to transfer into the user interface design. These `Tools` should follow generally accepted usability practices or state where they differ if this is a novel variation.

The `Tools` will be defined through three components, the system, application and elements. These represent the three separable areas which will normally be treated in a similar fashion throughout most environment designs. That is the system has its own paradigms which are broader than the application currently in use, the application itself will have a range of specific functional and visual traits and the interface elements themselves will have a combination of novel, application and system influenced representations.

***Consider next*:** `Users, Tasks, ToolsOpenFile, ToolsCloseFile`

***Task Efficiency*:** These must allow `Users` to complete `Tasks` within their current workspace wherever available and with the minimum amount of effort required.

***Reuse*:** Where possible existing tool paradigms from existing systems or the real world should be applied so that `Users` can reuse existing effort and knowledge. This should not be adhered to if the resulting `Tool` is more nostalgic than functional.

***User-Computer Communication*:** Facilitate collaboration between humans and computers.

***Robustness*:** All `Tools` should provide the necessary internal constraints and appropriate feedback for errors or options.

***Flexibility*:** Where possible the `Tools` should have multiple activation or interaction styles which cater for the differing levels of `Users` ability. This is often presented in the form of shortcuts for `UsersAdvanced` and Wizards for `UsersNovice` who may be less skilled.

8.3 ToolsOpenFile

Name: `ToolsOpenFile`

Context: This is the mechanism through which `Users` can retrieve files or datasets in order to further interact with them. While a similar action could be used for creating new files (ToolsNewFile) this is outside the scope of this experiment.

Consider these patterns first: `CoreInterface, Tools, Users.`

Problem: `Users` need to be able to easily find a specific file and activate it for editing or review.

Forces: The broader system will be unknown to the user in this experiment and therefore no existing paradigm of system level tools will exist. As this is the case `Users` will revert to expectations of WIMP (Hinckley, 1996) style or real world interfaces as they will have had at minimum some exposure to these.

There will possibly be many files to choose from, as such there must be a simple way for the `Users` to determine the difference between files.

Solution: Provide a tool which allows `Users` simple access to the file system through a predictable fashion which has clear differentiation between the files via various methods. From current operating systems we can draw differences such as filename, date of modification, file type and size (not as relevant to this experiment). There should be a method to allow the files displayed to be sorted to make selection easier.

Consider next: `Tasks, ToolsCloseFile, ToolsSelectObject`

Task Efficiency: To reduce load the system should first display the file system section which was last viewed and with the same sorting as used previously. This prevents repetitive actions being required when multiple opens are required.

Reuse: The selection should use the `ToolsSelectObject` standards for indication.

User-Computer Communication**:** The tool action should provide clear feedback that the sequence has been initiated and that opening of a file has occurred. For initiation this can be achieved through the use of a dialog box, flyout window or any strongly contrasting element which indicates that the functional mode has changed. When a file is opened, close the file selection environment and re-enforce the transition with some visual indicator such as a loading bar or similar.

Robustness**:** Minimise the options available through this task to decrease the possibility of error. Provide just what is needed, that is the options for selection and sorting and actioning, the opening of the selected file should be a separate action to the selection. There should also be a method to cancel the opening of a file.
If the opening of a file discards the data which is currently loaded then there must be an alert of some kind which indicates this.

Flexibility**:** The differentiation between differing file should be manipulable so that `Users` can sort based on differing parameters. There should be more than one method of making a selection (this will be limited by the available input devices).

8.4 ToolsCloseFile

Name: `ToolsCloseFile`

Context: This is the mechanic through which `Users` can close a data set once they have completed the intended `Tasks` or after choosing not to complete any Tasks.

Consider these patterns first: `CoreInterface, ToolsOpenFile, Users`

Problem: `Users` need to be able to easily close a file once they have completed working with a dataset this should also include a catch so the data can be saved if this action has not been taken.

Forces: `Users` will use this mechanic after having some exposure to the experimental system and will be looking for tool operations as they have so far experienced. The `Users` will most likely be looking to open another file or dataset or return to the editing environment, once the current file has been closed.

Solution: Provide at tool which allows `Users` a simple function through which they can close the file. From existing operating systems we can draw references such as a close button or menu command which is readily accessible. The process will likely be a common action and should be easily recognisable, fast to complete and predictable.

Consider next: `Tasks, ToolsSelectObject`

Task Efficiency: To reduce load, the mechanic should be able to be completed in a single action when appropriate, that is when and `UsersAdvanced` uses this tool with an understanding of the outcome. This may take more steps to complete if using a secondary method to invoke the action, a menu for instance, and if there is the need for a feedback loop.

Reuse: The tool should be consistent throughout the system and operate the same irrespective of the application type.

User-Computer Communication: The tool action should provide clear feedback that the sequence has been initiated and that closing of a file has occurred. For initiation this can be achieved through the use of a reducing window or any strongly contrasting element which indicates that the functional mode has changed. When a file is closed if the file has change but has not yet been saved then there should be a feedback catch. This feedback would be delivered via some sort of prompt, which would provide at a minimum the opportunity to cancel the action or to continue a third option of save and close would improve this communication.

Robustness**:** Minimise the options available through this task to decrease the possibility of error. The use of feedback as described above should be in keeping with the other `Tools` feedback, i.e. if the closing of a file discards the data which is currently loaded then there must be an alert of some kind which indicates this.

Flexibility**:** There should be more than one method of initiating this function, this could be buttons, menu, aural or gestural allowing differing `Users` to choose (this will be limited by the available input devices).

8.5 ToolsApplicationMenu

Name: `ToolsApplicationMenu`

Context: The name menu usually implies a listing item though this is not necessarily the case in AR and VR environments. A menu in this sense is the tool which allows `Users` to access a collection of choices of action or selection. Some of these have been discussed previously in this document.

Consider these patterns first: `Tools, Users, Tasks`

Problem: `Users` need to be presented with a collection of options for interaction and once a decision is made, need a process through which to commit that interaction. This differs from but can be combined with, contextual interaction prompts which occur as `Users` interact with objects.

Forces: Menus usually contain collection of regularly accessed information and so need to be easily accessible. The use of a menu removes focus, both system and human, from the current higher Tasks. Menus can contain more options than that which can easily be displayed within the given resolution and may contain varied levels of nested information.

Solution: The solution must include a collection of options which can be easily distinguished from each other and accessed in a predictable sequence. The location of the menu may vary depending on the environmental context but the method to access it must be consisted such as a floating button which reveals it, a specific gesture or voice command or a method of determining proximity to a hotspot or actionable item which is more appropriate for element menus than application wide menus.

Menus should provide for submenus or nested information in a consistent fashion and should have indicators which highlight the availability of further options and hint at the method for obtaining the further information.

Consider next: `ToolsApplicationSelectObject`

***Task Efficiency*:** The menu needs to be easily manipulable and within reach at all times. This does not mean that there is a need to be visible at all times but once initiated the process of use should be non-exhaustive.

***Reuse*:** The menu should make use of the common metaphors for selection and indication. After having used a menu once, the `Users` should reasonably expect all further menus should operate the same way, at minimum through the application and if possible through out the system.

***User-Computer Communication*:** The menu should provide at minimum feedback on the option currently being assessed for action and indication that a selection has been made and action is being taken. In a WIMP environment this is as simple as highlighting the current preselection option and collapsing the menu with audible feedback on selection, these base responses can be usefully transferred to the 3D environments.

***Robustness*:** If the selection option has a permanent affect on the current dataset then an alert or confirmation can be used along with an undo function which provides a return to the existing state prior to the action taken. The menu needs to have an exit function to allow `Users` to leave the menu environment and return focus to the previous environment or task.

***Flexibility*:** Shortcuts, single actions which bypass a lengthier menu navigation process, can be offered to certain common use options, these may be in the form of key commands, gestures or voice commands depending on the inputs available.

8.6 ToolsApplicationSelectObject

Name: `ToolsApplicationSelectObject`

Context: This is the tool or mechanism through which the `Users` can choose between objects or the appropriate sub parts of these. The process of selecting finer objects or elements is covered in `ToolsSelectObject`.

Consider these patterns first: `Tools, Users, Tasks`

Problem: During the use of the application the User will be required to make selections at the application level which may differ from the process of element based selection within the particular application function. This may occur when selecting a menu system or a major interface component.

Forces: The selection will need to draw focus from the current activity, at first this will not be the complete focus but should be an indicator the focus is to shift, such as an initial highlight which differs from the treatment of the active component.

The selection technique will to some extent be dependant on the input devices used and may be multi-modal.

Solution: The selection required for application level interactions is usually completed at a local level (Mine, 1995). This is typically achieved by the `Users` moving their selection device or pointer towards the target object which initiates some form of feedback when in contact or proximity to the pointer. This feedback can include any combination of visual change (highlight, colour shift), a positional shift, an auditory notification or revealing additional structural elements such as vertex handles. This conditional change must then revert if the `Users` decide not to select this object.

If selection does occur then the shift in focus occurs (`ToolsFocusObject`) and the component previously selected loses focus as the currently selected object gains focus.

Consider next: `ToolsSelectObject`

Task Efficiency**:** The nature of the change of each component on pre-selection must not be permanent and must revert when `Users` move away or complete their selection so that there is no lag in the selection process across multiple components. The change indicating selection must not be an extended process which requires `Users` to wait before moving to the next selection or be distracted from the next selection.

Reuse**:** The process needs to be consistent for all application level components.

User-Computer Communication**:** Feedback should be concise and direct, that is there should be zero (or near too) lag between when the `Users` expect a conditional change and when it occurs.

Robustness**:** The ability to shift the current selection should be as efficient as possible within the system capabilities as this function should become transparent to the user. As such this shift in current selection should be easily reverted, this is usually enabled simply by allowing the simple selection of the last object again without barrier.

Flexibility**:** To accommodate `Users` as their ability and familiarity increases there should be functions considered which allow for faster switching of selection such as shortcuts or combination actions which compliment the current activity, be it gestural, aural or prop based.

8.7 ToolsFocusObject

Name: `ToolsFocusObject`

Context: This is the tool or mechanism through which the `Users` are shown the current selection of the choice made between objects or the appropriate sub parts of these. The process of selecting objects or elements is covered in `ToolsSelectObject and ToolsApplicationSelectObject`.

Consider these patterns first: `Tools, Users, Tasks, ToolsApplicationSelectObject, ToolsSelectObject`

Problem: During the process of selecting an object or element for manipulation or activation, the user requires some feedback from the interface. This must first indicate that the object is available for selection and then that it has been selected.

Forces: The change in selection will need to draw focus from the current activity, at first this will not be the complete focus but should be an indicator the focus is to shift, such as an initial highlight which differs from the treatment of the active component.

Solution: Typically achieved by the `Users` moving their selection device or pointer towards the target object, this initiates some form of feedback when in contact or proximity to the pointer. This feedback can include any combination of visual change (highlight, colour shift), a positional shift, an auditory notification or revealing additional structural elements such as vertex handles. This conditional change must then revert if the `Users` decide not to select this object.

Once selected the object is actionable and as a result should appear to be in an active state. This in turn implies that objects should have an inactive state which is their natural state prior to gaining focus.

It is essential that only one object or specified grouping of objects have focus at any one time.

Consider next: `ToolsSelectObject`

Task Efficiency**:** The nature of the change of each component on pre-selection must not be permanent and must revert when `Users` move away or complete their selection so that there is no lag in the selection process across multiple components. The change indicating selection must not be an extended process which requires `Users` to wait before moving to the next selection or be distracted from the next selection.

Reuse**:** The process needs to be consistent for all components.

User-Computer Communication**:** Feedback should be concise and direct, that is there should be zero (or near too) lag between when the `Users` expect a conditional change and when it occurs.

Robustness**:** The ability to shift the current focus should be as efficient as possible within the system capabilities as this function should become transparent to the user. As such, this shift in current focus should be easily reverted, this is usually enabled simply be allowing the simple selection of the last object again without barrier.

Flexibility**:** Focus is not very flexible, it is singular in it nature, that is there can be only one object in focus at one time.

8.8 ToolsSelectObject

Name**:** `ToolsSelectObject`

Context**:** This is the tool or mechanism through which the `Users` can choose between object elements or the appropriate sub parts of these.

Consider these patterns first**:** `Tools, Users, Tasks, ToolsApplicationSelectObject`

Problem**:** During the use of the application the `Users` will be required to make selections at the object and sub-object level, that is the process of element based selection within the particular application function. This may occur when selecting a specific option within a menu, the selection of an object for manipulation or the selection of control points on specific Tools.

Forces**:** The selection will need to draw focus from the current activity, at first this will not be the complete focus but should be an indicator the focus is to shift, such as an initial highlight which differs from the treatment of the active component.

The selection technique will to some extent be dependant on the input devices used and may be multi-modal.

Solution**:** The selection required for element level interactions may be completed at both a local level (Mine, 1995) and at-a-distance. This is typically achieved by the `Users` moving their selection device or pointer towards the target object which initiates some form of feedback when in contact or proximity to the pointer. This feedback can include any combination of visual change (highlight, colour shift), a positional shift, an auditory notification or revealing additional structural elements such as vertex handles. This conditional change must then revert if the `Users` decide not to select this object. During selections made at-a-distance this pointer may be replaced with a range of devices such as ray casters and gaze directed selection.

If selection does occur then the shift in focus occurs (`ToolsFocusObject`) and the component previously selected loses focus as the currently selected object gains focus.

This element level selection differs from the application level in that there may be more variation in the feedback functions due to the contextual need of the element being selected. For instance, the selection of an option within a list like menu collection would be treated differently than the selection of a building within an architectural VR environment. While the base treatment would be similar, pre-selection feedback followed by shift in focus on selection, the building may require that a more complex set of nested tools are available on pre-selection allowing for sub-selection of elements for specific manipulation.

Consider next: `ToolsRotateObject, ToolsColourObject, ToolsScaleObject,`

Task Efficiency: The nature of the change of each component on pre-selection must not be permanent and must revert when `Users` move away or complete their selection so that there is no lag in the selection process across multiple components. The change indicating selection must not be an extended process which requires `Users` to wait before moving to the next selection or be distracted from the next selection.

Reuse: The process needs to be consistent for all elements which have similar function. All selection types however should have a common base method of interaction and feedback.

User-Computer Communication: Feedback should be concise and direct, that is there should be zero (or near too) lag between when the `Users` expect a conditional change and when it occurs.

Robustness: The ability to shift the current selection should be as efficient as possible within the system capabilities as this function should become transparent to the user. As such this shift in current selection should be easily reverted, this is usually enabled simply be allowing the simple selection of the last object again without barrier.

Flexibility: To accommodate `Users` as their ability and familiarity increases there should be functions considered which allow for faster switching of selection such as shortcuts or combination actions which compliment the current activity, be it gestural, aural or prop based.

8.9 ToolsRotateObject

Name: `ToolsRotateObject`

Context: This tool provides the user a mechanism for rotating the selected object through a specific range around a specific axis.

Consider these patterns first: `ToolsSelectObject, Tasks`

Problem: Via this mechanic `Users` will need to rotate objects about the x, y or z axis with accuracy to the degree. The tool should provide feedback of the current rotation and an opportunity to cancel the rotation which is underway.

Forces: The rotation will need to be completed one axis at a time at minimum. The viewpoint may not allow for accurate visual feedback when regarding the object itself.

Solution: The preferred execution allows for both rotate by feel and numeric input. That is there should be a mechanism through which `Users` can rotate the object manually (less accurate more feel) and numerically (precise, no feel). There needs to be a function for switching or locking to the specific axis of rotation.

Consider next: `, ToolsColourObject, ToolsScaleObject, Tasks`

Task Efficiency: Provide the ability to switch which axis quickly and access to activate the tool should be consistent with other available Tools.

Reuse: The element should use selection and manipulation techniques which are common across the `Tools` for the `CoreInterface`.

User-Computer Communication: Visual feedback of the specific rotation should be in real time with the initiation of the command.

Robustness: Provide at minimum a cancel function which will return the object to it’s state before engaging the tool. At best this will provide multiple “Undo’s” where `Users` can revert to the last rotation in progressively moving back through the rotation history since the tool was engaged, this would allow for partial corrections in a multistage rotation.

Flexibility**:** Access to shortcuts could be made available which would allow for faster access to the tool and could be extended to improve the accuracy of manual rotations such as constraining the rotation to specific angles throughout the full degree of freedom, that is, 15, 30 and 45 degree increments.

8.10 ToolsColourObject

Name: `ToolsColourObject`

Context: This tool provides the user a mechanism for altering the colour of the selected object through a specific range within the available colour gamut. This may be greyscale, 8, 16, 24 or 32 bit colour and so on as determined by the system environment.

Consider these patterns first: `ToolsSelectObject, Tasks`

Problem: Via this mechanic `Users` will need to vary the colour of objects with an accuracy which reflects the standard incremental unit for the colour system in place, i.e. through 256 increments per channel for 8-bit based colour systems which most display devices utilise. The tool should provide feedback of the current colour per channel and an opportunity to cancel the variation which is underway.

Forces: The rotation will need to be completed one channel at a time at minimum. The viewpoint may not allow for accurate visual feedback when regarding the object itself, if this is the case then a colour swatch should be visible when manipulation occurs.

Solution: The preferred execution allows for both colourations by feel and numeric input. That is there should be a mechanism through which `Users` can colour the object manually (less accurate more feel) and numerically (precise, no feel). A colour swatch should be visible which shows the before and after colour of the object.

Consider next: `, ToolsRotateObject, ToolsScaleObject, Tasks`

Task Efficiency: Provide the ability to switch colour channels quickly and access to activate the tool should be consistent with other available Tools.

To increase efficiency and consistency of colours, a set of colour swatches or presets can be made available which would allow for simple selection of the same option many times. If this is available then `Users` should be able to save their own colours to this selection set.

Reuse**:** The element should use selection and manipulation techniques which are common across the `Tools` for the `CoreInterface`.

User-Computer Communication**:** Visual feedback of the colour change should be in real time with the initiation of the command.

Robustness**:** Provide at minimum a cancel function which will return the object to it's state before engaging the tool. At best this will provide multiple "Undo's" where `Users` can revert to the last variation and progressively move back through the interaction history since the tool was engaged, this would allow for partial corrections in a multistage manipulation.

Flexibility**:** Access to shortcuts could be made available which would allow for faster access to the tool and could be extended to improve the accuracy of manual manipulations such as constraining the variations to specific increments throughout the full degree of freedom of variation.

8.11 ToolsScaleObject

Name: `ToolsScaleObject`

Context: This tool provides the user a mechanism for scaling the selected object through a specific range around a specific axis.

Consider these patterns first: `ToolsSelectObject, Tasks`

Problem: Via this mechanic `Users` will need to scale objects about the `x`, `y` or `z` axis with accuracy to the unit of measure within the system or by percentage. The tool should provide feedback of the current scaling and an opportunity to cancel the scaling which is underway.

Forces: The scale will need to be completed one axis at a time at minimum. The viewpoint may not allow for accurate visual feedback when regarding the object itself.

Solution: The preferred execution allows for both scaling by feel and numeric input. That is there should be a mechanism through which `Users` can scale the object manually (less accurate more feel) and numerically (precise, no feel). .

Consider next: `, ToolsRotateObject, ToolsColourObject, Tasks`

Task Efficiency: Provide the ability to switch which axis quickly and access to activate the tool should be consistent with other available Tools.

To increase efficiency and consistency of scaling a presets can be made available which would allow for simple selection of the same option many times. This preset can simply be the last used input values.

Reuse: The element should use selection and manipulation techniques which are common across the `Tools` for the `CoreInterface`.

User-Computer Communication: Visual feedback of the scale change should be in real time with the initiation of the command.

Robustness: Provide at minimum a cancel function which will return the object to it's state before engaging the tool. At best this will provide multiple "Undo's" where `Users` can revert to the last variation and progressively move back through the interaction history since the tool was engaged, this would allow for partial corrections in a multistage manipulation.

Flexibility**:** Access to shortcuts could be made available which would allow for faster access to the tool and could be extended to improve the accuracy of manual manipulations such as constraining the variations to specific increments throughout the full degree of freedom of variation.

8.12 Tasks

***Name*:** `Tasks`

***Context*:** `Users` are presented a range of scenarios in which they need to achieve some end goal, that is, complete a specific task. This is accomplished by utilising the available and appropriate `Tools`.

***Consider these patterns first*:** `CoreInterface, Tools, Users`

***Problem*:** The need to achieve the specific end goal, in this experiment is multilayered in that there are a series of `Tasks` for `Users` to complete each with their own goal and the overarching task which will allow for the adequate testing of the interface which has been developed.

***Forces*:** The `Users` will need to use only the `Tools` provided to achieve the goals stated. The `Users` will have varied levels of ability and prior knowledge of how to complete each of the `Tasks`.

***Solution*:** Provide clear definition of the end goal and any surrounding parameters which need to be considered to execute the task. Where `Tasks` are multi-step, present them in a context which indicates this unless the discovery of the process is part of the task itself.

***Consider next*:** `Tools, Users, TasksOpenFile`

***Task Efficiency*:** Where possible the `Tasks` should be presented in an order which promotes efficiency rather than rework.

***Reuse*:** Ensure that `Users` can reuse existing effort and knowledge by providing similar clues to the end goals and by allowing the use of similar `Tools` to complete multiple `Tasks`.

***User-Computer Communication*:** Provide clear indication that the `Tasks` have been completed correctly or not by stating clear end goals.

***Robustness*:** If `Tasks` are attempted and not completed then there must be the facility to attempt them again.

Flexibility**:** Where multiple options for completion exist, do not be so descriptive in the `Tasks` presentation that the `Users` are forced to follow a step by step process to execute the `Tasks`.

8.13 TasksOpenFile

Name**:** `TasksOpenFile`

Context**:** `Users` are presented the system and application and required to perform a specific task. This is accomplished by utilising the available and appropriate Tools.

Consider these patterns first**:** `CoreInterface, Tools, Users`

Problem**:** `Users` are required to retrieve files or datasets in order to further interact with them.

Forces**:** There may be many files to choose from, so the specific file should be stated clearly in the task assignment.

Solution**:** The task should be presented as such:

Using either the `ToolsOpenFile` or another method known to you please open the file name FileName so that it is available for editing.

Where FileName is the file specifically required for this experiment stage.

Consider next**:** `TasksCloseFile, ToolsOpenFile`

Task Efficiency**:** This task should be presented each time a new file is required and before any manipulation `Tasks` so the user does not begin manipulating data before opening the required file.

Reuse**:** Each time this is presented the language should be similar.

User-Computer Communication**:** Providing clear indication that `Tasks` have been completed should be handled by the `Tools` in mechanical terms. This can be accompanied by a closing statement of acknowledgement when the `Tasks` have been completed.

Robustness**:** If this `Tasks` request occurs when a file is already open then provide additional information on how to deal with the files already open. Either by stating or implying that more than one file can be opened at once or by prompting the closure of the first file before opening the second.

***Flexibility*:** If `Users` are aware of available shortcuts or alternate paths then they should be allowed to use them.

8.14 TasksCloseFile

Name: `TasksCloseFile`

Context: `Users` are presented the system and application where at least one file or dataset is active and required to perform a specific task. This is accomplished by utilising the available and appropriate Tools.

Consider these patterns first: `CoreInterface, Tools, Users`

Problem: `Users` are required to close files or datasets in order to conclude interaction with them.

Forces: There may be too many files to choose from so the specific file should be stated clearly in the task assignment.

Solution: The task should be presented as such:

Using either the `ToolsCloseFile` or another method known to you please close the file name FileName so that it is no longer available for editing.

Where FileName is the file specifically required for this experiment stage.

Consider next: `TasksOpenFile, ToolsCloseFile`

Task Efficiency: This task should be presented each time a file is no longer required for manipulation Tasks.

Reuse: Each time this is presented the language should be similar.

User-Computer Communication: Providing clear indication that `Tasks` have been completed should be handled by the `Tools` in mechanical terms. This can be accompanied by a closing statement of acknowledgement when the `Tasks` have been completed.

Robustness: If there are multiple files open, ensure that the name is clearly stated and if the data set requires saving before closure then ensure that this is requested at the same time.

***Flexibility*:** If `Users` are aware of available shortcuts or alternate paths then they should be allowed to use them.

8.15 TasksApplicationAccessMenu

Name: `TasksApplicataionAccessMenu`

Context: `Users` are presented the system and application where at least one file or dataset is active and are required to access the available and appropriate `Tools` via access to a menu.

Consider these patterns first: `Tools, TasksOpenFile, TasksCloseFile`

Problem: `Users` are required to choose from a selection of options presented in a collection or menu. This collection may be accessed through differing mechanics and may be invoked via differing methods.

Forces: There may be many options to choose from so the specific goal should be stated clearly when the task is assigned. `Users` may not be aware of how to access the menu and this should be a simple self discovery process.

Solution: The task should be presented as such:

To perform the DesiredGoal, choose the DesiredOption from the appropriate menu.

Where the DesiredGoal and DesiredOption are the task goal and the menu option required respectively.

Consider next: `TasksApplicataionSelectObject`

Task Efficiency: This task should be presented each time a choice of options from a collection is required.

Reuse: Each time this is presented the language should be similar.

User-Computer Communication: Providing clear indication that `Tasks` have been completed should be handled by the `Tools` in mechanical terms. This can be accompanied by a closing statement of acknowledgement when the `Tasks` have been completed.

***Robustness*:** If there are multiple menus available, ensure that the one required is clearly stated.

***Flexibility*:** If `Users` are aware of available shortcuts or alternate paths then they should be allowed to use them.

8.16 TasksApplicationSelectObject

Name: `TasksApplicataionSelectObject`

Context: `Users` are presented the system and application where at least one file or dataset is active, and required to perform alternate selections of the available and appropriate `Tools`.

Consider these patterns first: `Tools, TasksOpenFile, TasksApplicataionAccessMenu`

Problem: `Users` are required to initiate a selection of specific `Tools` or menus at the application level rather than the individual element level which is dealt with by the `TasksSelectObject` pattern. The available `Tools` may be accessed through differing mechanics and may be invoked via differing methods.

Forces: There may be many `Tools` to choose from so the specific goal should be stated clearly when the task is assigned. `Users` may not be aware of how to access the appropriate `Tools` and this should be a simple self discovery process.

Solution: The task should be presented as such:

From the available tools, choose the DesiredOption from the appropriate menu or currently accessible `Tools`.

Where the DesiredOption is option required to complete the selection task.

Consider next: `TasksSelectObject, TasksApplicataionAccessMenu`

Task Efficiency: This task should be presented each time a choice of options from a collection is required.

Reuse: Each time this is presented the language should be similar.

User-Computer Communication: Providing clear indication that `Tasks` have been completed should be handled by the `Tools` in mechanical terms. This can be accompanied by a closing statement of acknowledgement when the `Tasks` have been completed.

Robustness**:** If there are multiple `Tools` available ensure that the one required is clearly stated.

Flexibility**:** If `Users` are aware of available shortcuts or alternate paths then they should be allowed to use them.

8.17 TasksSelectObject

Name: `TasksSelectObject`

Context: `Users` are presented the system and application where at least one file or dataset is active and required to perform alternate `Tasks` using the available and appropriate elements within the currently available dataset.

Consider these patterns first: `TasksOpenFile, TasksApplicataionSelectObject, ToolsSelectObject`

Problem: `Users` are required to initiate the selection of a specific object or element through direct selection. This task is performed with the intention of further manipulation once the selection has been made.

Forces: The selection will need to draw focus from the current object, at first this will not be the complete focus but should be an indicator the focus is to shift, such as an initial highlight which differs from the treatment of the active object.

The selection technique will to some extent be dependant on the input devices used and may be multi-modal.

Solution: The task should be presented as such:

From the available elements or objects, choose the DesiredObject so that the DesiredManipulation can be performed.

Where the DesiredObject is object or element required to complete the selection task and DesiredManipulation are the following `Tasks` to be performed.

Consider next: `TasksRotateObject, TasksColourObject, TasksScaleObject,`

Task Efficiency: This task should be presented each time a choice of options from a collection is required.

Reuse: Each time this is presented the language should be similar.

***User-Computer Communication*:** Providing clear indication that `Tasks` have been completed should be handled by the `Tools` in mechanical terms. This can be accompanied by a closing statement of acknowledgement when the `Tasks` have been completed.

***Robustness*:** If there are multiple objects available for selection each with specific sub-elements which may all be available for selection ensure that the one required is clearly stated.

***Flexibility*:** If `Users` are aware of available shortcuts or alternate paths then they should be allowed to use them.

8.18 TasksRotateObject

Name**:** `TasksRotateObject`

Context**:** `Users` are presented the system and application where at least one file or dataset is active and required to perform alternate `Tasks` using the available and appropriate elements within the currently available dataset.

Consider these patterns first**:** `ToolsSelectObject, ToolsRotateObject`

Problem**:** `Users` are required to select an object, `TasksSelectObject`. Once the selection has been made `Users` are require to perform a specific rotation or series of rotations of the object. This rotation can be presented either as a degree specific precise rotation or a general rotation such as “turn the object upside down”.

Forces**:** `Users` will need to first make an appropriate selection and access the appropriate `Tools` for the task.

The selection technique will to some extent be dependant on the input devices used and may be multi-modal.

Solution**:** The task should be presented as such:

From the available elements or objects, choose the DesiredObject so that the object can be rotated. Once selected, rotate the object through the DesiredRotation using the rotation tool.

Where the DesiredObject is object or element required to complete the task and DesiredRotation is the rotation to be performed, either specific or general.

Consider next**:** `TasksSelectObject, TasksColourObject, TasksScaleObject,`

Task Efficiency**:** This task should be presented each time a rotation is required.

Reuse**:** Each time this is presented the language should be similar.

User-Computer Communication**:** Providing clear indication that `Tasks` have been completed should be handled by the `Tools` in mechanical terms. This can be accompanied by a closing statement of acknowledgement when the `Tasks` have been completed.

Robustness**:** If there are multiple objects available for selection each with specific sub-elements which may all be available for selection ensure that the one required is clearly stated.

Flexibility**:** If `Users` are aware of available shortcuts or alternate paths then they should be allowed to use them.

8.19 TasksColourObject

Name: `TasksColourObject`

Context: `Users` are presented the system and application where at least one file or dataset is active and required to perform alternate `Tasks` using the available and appropriate elements within the currently available dataset.

Consider these patterns first: `ToolsSelectObject, ToolsColourObject`

Problem: `Users` are required to select an object, TasksSelectObject. Once the selection has been made `Users` are require to change the colour of the object or element. This colouring can be presented either as a numerically specific precise modification or a general modification such as “a shade of blue”. `Users` can also use one of the predefined colour swatches if available.

Forces: `Users` will need to first make an appropriate selection and access the appropriate `Tools` for the task.

The selection technique will to some extent be dependant on the input devices used and may be multi-modal.

Solution: The task should be presented as such:

From the available elements or objects, choose the DesiredObject so that the object can be coloured. Once selected change the objects colour to DesiredColour using the colouring tool.

Where the DesiredObject is the object or element required to complete the task and DesiredColour is the colour definition for the shift to be performed, either specific or general.

Consider next: `TasksSelectObject, TasksRotateObject, TasksScaleObject,`

Task Efficiency: This task should be presented each time a colour shift is required.

Reuse**:** Each time this is presented the language should be similar.

User-Computer Communication**:** Providing clear indication that `Tasks` have been completed should be handled by the `Tools` in mechanical terms. This can be accompanied by a closing statement of acknowledgement when the `Tasks` have been completed.

Robustness**:** If there are multiple objects available for selection each with specific sub-elements which may all be available for selection ensure that the one required is clearly stated.

Flexibility**:** If `Users` are aware of available shortcuts or alternate paths then they should be allowed to use them.

8.20 TasksScaleObject

Name**:** `TasksScaleObject`

Context**:** `Users` are presented the system and application where at least one file or dataset is active and required to perform alternate `Tasks` using the available and appropriate elements within the currently available dataset.

Consider these patterns first**:** `ToolsSelectObject, ToolsScaleObject`

Problem**:** `Users` are required to select an object, TasksSelectObject. Once the selection has been made `Users` are required to change the scale of the object or element. This scaling can be presented either as a numerically specific precise modification or a general modification such as “bigger or smaller”.

Forces**:** `Users` will need to first make an appropriate selection and access the appropriate `Tools` for the task.

The selection technique will to some extent be dependant on the input devices used and may be multi-modal.

Solution**:** The task should be presented as such:

From the available elements or objects, choose the DesiredObject so that the object can be scaled. Once selected change the objects scale to DesiredScale using the colouring tool.

Where the DesiredObject is object or element required to complete the task and DesiredScale is the scale definition for the shift to be performed, either specific or general.

Consider next**:** `TasksSelectObject, TasksRotateObject, TasksColourObject,`

Task Efficiency**:** This task should be presented each time a scaling is required.

Reuse**:** Each time this is presented the language should be similar.

***User-Computer Communication*:** Providing clear indication that `Tasks` have been completed should be handled by the `Tools` in mechanical terms. This can be accompanied by a closing statement of acknowledgement when the `Tasks` have been completed.

***Robustness*:** If there are multiple objects available for selection, each with specific sub-elements which may all be available for selection ensure that the one required is clearly stated.

***Flexibility*:** If `Users` are aware of available shortcuts or alternate paths then they should be allowed to use them.

8.21 Users

Name**:** `Users`

Context**:** This is the collection of people who are to actively use the system. They have varied familiarity with computers and may have been exposed to a range of applications, interfaces, input devices and task types.

Consider these patterns first**:** `CoreInterface, Tools, Tasks`.

Problem**:** A collection of users are required to provide feedback from range of viewpoints which may or may not have been exposed to similar experiments or interface developments. This allows the experiential feedback to be provided from multiple perspectives and without the bias brought to the interface by specific past experience.

Forces**:** The user types are ranged into novice, intermediate and advanced to ensure there are a range of user types. It is best for testing to have an even spread of these types.

Solution**:** Subjects should come from a range of backgrounds, some with minimal exposure, some who use computers regularly and some using computing professionally, as a hobbyist or computer game enthusiasts and with a broad exposure to various system paradigms. Predisposition to any particular environment across the collection of `Users` could reduce the accuracy of the results.

Consider next**:** `UsersAdvanced, UsersIntermediate, UsersNovice`

8.22 UsersAdvanced

Name**:** `UsersAdvanced`

Context**:** This user type is very familiar with computers and has been exposed to a range of applications, interfaces, input devices and task types. This user has a developed understanding of computing and has experience with self learning and system adaptation.

Consider these patterns first**:** `CoreInterface, Users`.

Problem**:** This user type is utilised to provide developed feedback from an informed viewpoint which may or may not have been exposed to similar experiments or interface developments. This allows the experiential feedback to be provided without the need to learn the fundamentals and with minimal adjustment to the specific environment.

Forces**:** The user type must have developed broad computing experience, i.e. multiple operating systems, display types and computing purpose not simply long term exposure to single or similar task types. This cross system usage will influence the knowledge transfer of interface and input types which will affect directly the responses and measured outcomes.

Solution**:** Subjects should come from a computing or related background, though not necessarily professionally as hobbyist and computer game enthusiasts may have a broad exposure to various system paradigms. This user type must be able to adapt to varied computing environments without an excessive predisposition to any particular environment which could remove the impartiality of the results.

Consider next**:** `UsersIntermediate, UsersNovice`

8.23 UsersIntermediate

Name**:** `UsersIntermediate`

Context**:** This user type is familiar with computers and has had some exposure to a range of applications, interfaces, input devices and task types. This user has an understanding of computing and has had some experience with self learning and system adaptation though this is limited. This user type encapsulates the majority of `Users` who would use computers on a daily or near daily basis and are familiar with general interface and application paradigms.

Consider these patterns first**:** `CoreInterface, Users.`

Problem**:** This user type is utilised to provide general feedback from a viewpoint which may or may not have been exposed to similar experiments or interface developments. This allows the experiential feedback to be provided with the need to revisit the fundamentals and with some adjustment to the specific environment.

Forces**:** The user type must have some computing experience, i.e. limited exposure to multiple operating systems, display types and computing purposes the bulk of which may be limited to long term exposure to single or similar task types.

Solution**:** Subjects should come from a background where computing is a common task, this may be through hobbyist and computer game activities. This user type must be able to recognise varied computing environments without necessarily understanding the transferred interaction immediately.

Consider next**:** `UsersAdvanced, UsersNovice`

8.24 UsersNovice

Name**:** `UsersNovice`

Context**:** This user type is aware of computers and has had limited exposure to a range of applications, interfaces, input devices and task types. This user has an underdeveloped understanding of computing and has had little to no experience with self learning and system adaptation. This user type encapsulates the range of `Users` who would user computers on an infrequent or assisted basis and are vaguely familiar with general interface and application paradigms.

Consider these patterns first**:** `CoreInterface, Users.`

Problem**:** This user type is utilised to provide general feedback from a viewpoint which would most likely not have been exposed to similar experiments or interface developments. This allows the experiential feedback to be provided with the need to develop the fundamentals and with total adjustment to the specific environment. This user will also provide the most direct feedback as to how intuitive the interface is when compared with the real world equivalents as the user will have limited frame of reference in the virtual environment.

Forces**:** The user type must have little or strictly limited computing experience, i.e. limited exposure to a single operating system, display type or computing purposes. Any experience may be limited to short term exposure to single or similar task types which may have been delivered in an assisted environment.

Solution**:** Subjects should come from a background where computing is not a common task. This user type is not required to be able to recognise varied computing environments and will usually draw on real world experiences when developing an understanding or interaction presented.

Consider next**:** `UsersAdvanced, UsersIntermediate`

9 Appendix B – Sample Test

9.1 Task list

Using the interface available perform the following tasks.

1. Open the file “RectangularPrisms”.
2. Select the largest of the three rectangular prisms.
3. Selecting the Rotation Tool from the appropriate menu
4. With this prism selected rotate it about the x-axis by 90 degrees using the rotation tool.
5. Save and close the file.

9.2 Alternate task list for further study examples

Using the interface available perform the following tasks.

1. Open the file “RectangularPrisms”.
2. Select the largest of the three rectangular prisms.
3. By selecting the Scaling Tool from the appropriate menu, scale this prism so that is roughly equal in size to the middle sized prism.
4. With this prism still selected rotate it about the x-axis by 90 degrees using the rotation tool.
5. Select the Colour Tool from the appropriate menu
6. Now select the smallest prism and change it’s colour to blue.
7. Save and close the file.

10 Appendix C – User Selection Test

Used to Determine the Experience Level of Participants

Please complete the following questionnaire and return to the project supervisor.

1. Are you familiar with using computers?

☐ yes / ☐ no

2. Do you own a computer?

☐ yes / ☐ no

3. Do you use computers for (check all that apply)

☐ personal reasons ☐ work/study related reasons

4. What type of applications do you use computers for (check all that apply)?

☐ word processing (Word, Word Perfect, etc.)

☐ graphics/drawing (CAD, Corel Draw, Photoshop, etc.)

☐ internet or email (Eudora, Firefox, Internet Explorer, Outlook etc.)

☐ programming/ simulation (C, VB, ASP, etc.)

☐ multimedia (Authorware, Director, Flash, etc.)

☐ spreadsheets (Excel, Delta Graph, etc.)

other _______________

5. Do you use the internet on a regular basis?

☐ yes / ☐ no

6. Have you used Virtual Reality or Augmented Reality software before?

☐ yes / ☐ no

If yes provide details ________________________

7. How would your rate your general computer skills

☐ poor ☐ below average ☐ average ☐ above average ☐ advanced

8. How would your rate you use of computer games, desktop or console.

☐ poor ☐ below average ☐ average ☐ above average ☐ advanced

9. Have you ever been involved in the design or development of software, multimedia or data visualisation?

☐ yes, complex applications ☐ yes, simple applications ☐ no

10.1 Answer Key for User Selection

Question	Novice	Intermediate	Advanced
1	No	Yes	Yes
2	No	Yes	Yes
3	Work/study	Personal/ work/study	Personal/ work/study
4	3 or less options	4 – 5 options	5 or more options
5	No	Yes	Yes
6	No	No	Yes
7	poor / below average	average	above average / advanced
8	poor / below average	average	above average / advanced
9	No	Yes simple	Yes complex

Users are placed in the category within which the majority of their answers fall. If the user falls within two categories evenly, the subject will be placed in the lower of the two categories.

11 Appendix D – Post Task Evaluation

To be given to each user at the completion of the tasks for each interface variation.

For completion by Experiment Supervisor

Date and Time	
User type	
Interface Used	
Task Combination	

For completion by Experiment Participant

Please complete the following questions by marking on the following scales which answer you feel is most appropriate.

1. The visual design of this interface was appealing.				
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

2. The visual design of this interface made the environment and tools easier to understand.				
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

3. I was able quickly recognise the functions of different tools and interface elements.				
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

4. The interface and its elements were easy to understand.				
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

5. I believe that the time I have spent doing similar tasks assisted in my understanding of this interface.				
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

6. The interface felt nice to use.				
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

7. I was able to accurately complete the set task				
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

8. The interface design assisted me to accurately complete the set task				
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

9. I was able to quickly complete the set task				
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

10. The interface design assisted me to quickly complete the set task				
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

11. I feel the use of colour in this design is makes the interface better to use.				
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

12. I feel the use of type in this design is makes the interface better to use.				
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

13. I feel the placement of elements in this design is makes the interface better to use.				
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

14. List some key words which you feel describe this interface.

____

____

____

____

15. Do you have any other comments about this interface?

____

____

____

____

Please return this to the Experiment Supervisor. Thank you for participating.

12 Bibliography

Alexander, C. (2001), *Pattern Language*, Website, Accessed: 21/02/06, http://www.patternlanguage.com/

Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I. and Angel, S. (1977), *A Pattern Language,* Oxford University Press, New York.

Apple Computer Inc. (2006), *Apple Human Interface Guidelines*, Website, Accessed: 15/08/06, http://developer.apple.com/documentation/UserExperience/Conceptual/OSXHIGuidelines/index.html#//apple_ref/doc/uid/20000957

Appleton, B. (2000), *Patterns and Software: Essential Concepts and Terminology*, Brad Appleton, Website, Accessed: 2/9/05, http://www.cmcrossroads.com/bradapp/docs/patterns-intro.html#Origins

Avery, B., Thomas, B. H., Velikovsky, J. and Piekarski, W. (2005), *Outdoor Augmented Reality Gaming on Five Dollars a Day*, Proceedings of AUIC2005,

Azuma, R. T. (1997), In *SIGGRAPH '97 Course Notes #30: Making Direct Manipulation Work in Virtual Reality*.

Barlow, S. T. (1999), *Spatial Knowledge Acquired Through Navigation In A Large-Scale Virtual Environment,* Doctor of Philosophy Psychology, North Carolina State University.

Barrass, S. (1998), *Auditory Information Design,* Doctor of Philosophy, ANU, Canberra.

BBC-NEWS (2004), *'Laser vision' offers new insights*, BBC NEWS, Website, Accessed: 30/04/04, http://news.bbc.co.uk/1/hi/technology/3647437.stm

Beck, K. and Cunningham, W. (1987), *Using Pattern Languages for Object-Oriented Programs*, OOPSLA-87 Workshop on the Specification and Design for Object-Oriented Programming, Technical Report No. CR-87-43

Ben Hajji, F. and Dybner, E. (1999), *3D Graphical User Interfaces,* Department of Computer and Systems Sciences, Stockholm University and The Royal Institute of Technology.

Billinghurst, M. and Savage, J. (1996), *Adding Intelligence to the Interface*, Proceedings of VRAIS '96.,

Bowman, D. A. (1999), *Interaction Techniques for Immersive Virtual Environments: Design, Evaluation, and Application,* Doctor of Philosophy Computer Science, Georgia Institute of Technology, Georgia.

Bowman, D. A. (2002), *Navigation in Virtual Environments*, Virginia Tech

Bowman, D. A., Kruijff, E., Joseph J. LaViola, J. and Poupyrev, I. (2001), *An Introduction to 3D User Interface Design*, Presence: Teleoperators and Virtual Environments*,* 10, 96-108.

Bowman, D. A., Kruijff, E., LaViola, J. J. J. and Poupyrev, I. (2004), *3D User Interfaces: Theory and Practice,* Addison Wesley Professional.

Bowman, D. A. and Wingrave, C. A. (2001), *Design and Evaluation of Menu Systems for Immersive Virtual Environments*, Proceedings of Proceedings of IEEE Virtual Reality, 2001, 149-156

Chen, C., Czerwinski, M. and Macredie, R. (2000), *Individual Differences in Virtual Environments - Introduction and Overview*, Journal of the American Society for Information Science*,* 51, 499-507.

Cognitive Science Laboratory (2006), *Wordnet*, Cognitive Science Laboratory, Princeton University, Accessed: 06/04/06, http://wordnet.princeton.edu/perl/webwn%3Fs%3Dmanipulation

Cruz-Niera, C., Sandin, D. and Defanti, T. (1993), *Surround Screen Projection-Based Virtual Reality*, Proceedings of SIGGRAPH'93, 135-142

Darken, R. P. and Sibert, J. L. (1993), *A Toolset for Navigation in Virtual Environments*, Proceedings of ACM User Interface Software & Technology, 157-165

Dick, P. K. and Frank, S. (2002), *Minority Report,* Spielberg, S., Twentieth Century Fox and Dreamworks

Döllner, J. and Hinrichs, K. (1998), *Interactive, Animated 3D Widgets*, Proceedings of Computer Graphics International CGI98, 278-286

Entertainment Software Association (ESA) (2005), *Top 10 Industry Facts*, Entertainment Software Association (ESA), Website, Accessed: 10/8/06, http://www.theesa.com/facts/top_10_facts.php

Ergen, F. F. (1996), *Effects of Interface Format, Feedback Style and System Lag on the Usability of Hand-Held Internet Controllers,* Master of Science Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg.

Everett, S. S., Wauchope, K. and Pérez-Quiñones, M. A. (1998), *Creating Natural Language Interfaces to VR Systems: Experiences, Observations and Lessons Learned*, Proceedings of VSMM98, 4th International Conference on Virtual Systems and Multimedia, 2, 469-474

FakeSpace Systems (2006), *FLEX and reFLEX Product Brochure*,

Foley, J. (1979), *A Standard Computer Graphics Subroutine Package*, Computers and Structures*,* 10, 141-147.

Frye, J., Yoder, J. and The-Hillside-Group (2005), *http://www.hillside.net/*, The Refactory, Inc., Website, Accessed: 2005/6, http://www.hillside.net/

Funkhouser, T., Jot, J.-M. and Tsingos, N. (2002), In *SIGGRAPH 2002 Course Notes*.

Gabbard, J., Swartzz, K., Richey, K. and Hix, D. (1999), *Usability Evaluation Techniques: A Novel Method For Assessing The Usability Of An Immersive Medical Visualization VE*, Proceedings of VWSIM'99, 165-170

Gabbard, J. L. (1997), *A Taxonomy of Usability Characteristics in Virtual Environments,* Virginia Polytechnic Institute and State University, Virginia.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. (1995), *Design Patterns: Elements of Reusable Object-Oriented Software,* Addison-Wesley Professional Computing Series.

Grönegress, C., Thomsen, M. R. and Slater, M. (2001), *Designing Intuitive Interfaces for Virtual Environments,* Masters in Computer Vision, Image Processing, Graphics and Simulation, University College London, London.

Hinckley, K. (1996), *Haptic Issues for Virtual Manipulation,* Doctor of Philosophy Computer Science, University of Virginia.

Hix, D., Swan, J., Gabbard, J., McGee, M., Durbin, J. and King, T. (1999), *User-Centered Design and Evaluation of a Real-Time Battlefield Visualization Virtual Environment*, Proceedings of IEEE Virtual Reality '99, 96-103

Hubbold, R., Murta, A., West, A. and Howard, T. (1993), *Design Issues for Virtual Reality Systems*, Proceedings of First Eurographics Workshop on Virtual Environments, 27 - 38

Hussey, A., MacColl, I. and Carrington, D. (2001), *Assessing Usability from Formal User-Interface Designs*, Proceedings of 13th Australian Software Engineering Conference (ASWEC'01), 40-48

Ingram, R. and Benford, S. (1995), *Improving the Legibility of Virtual Environments*, Proceedings of 2nd Eurographics Workshop on Virtual Environments, 211-223

LaViola, J. (1999), *Whole-Hand and Speech Input in Virtual Environments,* Master's Thesis, Brown University.

Lecuyer, A., Megard, C., Burkhardt, J.-M., Lim, T., Coquillart, S., Coiffet, P. and Graux, L. (2002), *The Effect of Haptic, Visual and Auditory Feedback on an Insertion Task on a 2-Screen Workbench*, Proceedings of Immersive Projection Technology Symposium (IPT), Leonard, B. and Everett, G. (1992), *The Lawnmower Man,* Leonard, B., New Line Cinema

Lindeman, R. W., Sibert, J. L. and Hahn, J. K. (1999a), *Hand-Held Windows: Towards Effective 2D Interaction in Immersive Virtual Environments (1999)*, In *IEEE Virtual Reality*, pp. 205-212.

Lindeman, R. W., Sibert, J. L. and Hahn, J. K. (1999b), *Towards Usable VR: An Empirical Study of User Interfaces for Immersive Virtual Environments*, Proceedings of CHI '99, 64-71

Lindeman, R. W. and Templeman, J. N. (2001), *Vibrotactile feedback for handling virtual contact in immersive virtual environments*, In *Usability Evaluation and Interface Design: Cognitive Engineering, Intelligent Agents and Virtual Reality*, (Ed, Smith, M. J., Salvendy, G., Harris, D., and Koubek, R.J.), pp. 21-25.

Lindeman, R. W., Templeman, J. N. and Sibert, J. L. (2001), *The Effect of 3D Widget Representation and Simulated Surface Constraints on Interaction in Virtual Environments*, Proceedings of IEEE Virtual Reality 2001, 141-147

Lindeman, R. W., Templeman, J. N., Sibert, J. L. and Cutler, J. R. (2002), *Handling of Virtual Contact in Immersive Virtual Environments: Beyond Visuals*, Virtual Reality*,* 6, 130-139.

Lisberger, S. and MacBird, B. (1982), *Tron,* Lisberger, S., Buena Vista Pictures

Mahemoff, M. J. and Johnston, L. J. (1998a), *Pattern Languages for Usability: An Investigation of Alternative Approaches*, Proceedings of Asia-Pacific Conference on Human Computer Interaction (APCHI) 98, 25-31

Mahemoff, M. J. and Johnston, L. J. (1998b), *Principles for a usability-oriented pattern language*, Proceedings of Australasian Computer Human Interaction Conference, 132 - 139

Martens, J.-B., Qi, W., Aliakseyeu, D., Kok, A. and van Liere, R. (2004), *Experiencing 3D Interactions in Virtual Reality and Augmented Reality*, Proceedings of EUSAI 2004,

McGee, M. K. (1998), *Assessing Negative Side Effects in Virtual Environments,* Master of Science in Industrial and Systems Engineering, Faculty of the Virginia Polytechnic Institute and State University, Blacksburg.

Meijer, P. B. L. (1992), *An Experimental System for Auditory Image Representations*, IEEE Transactions on Biomedical Engineering*,* 39, 112-121.

Meijer, P. B. L. (2006), *Sensory Substitution - Vision Substitution*, Meijer, Peter B.L., Website, Accessed: 17/04/06, http://www.seeingwithsound.com/sensub.htm

Mine, M. R. (1995), *Virtual Environment Interaction Techniques*, Department of Computer Science, University of North Carolina, Chapel Hill, NC 27599-3175, TR95-018

Mynatt, E. D. (1995), *Transforming Graphical Interfaces into Auditory Interfaces,* Doctor of Philosophy Computer Science, Georgia Institute of Technology, Georgia.

Nielsen, J. (2000), *Designing Web Usability: The Practice of Simplicity,* New Riders Publishing, Indianapolis.

Noma, H., Miyasato, T., Kitamura, Y. and Kishino, F. (1996), *Haptic and Visual Feedback for Manipulation Aid in a Virtual Environment*, Proceedings of Fifth Annual Symposium on Haptic Interface for Virtual Environments and Teleoperated Systems,

Pierce, J. S., Pausch, R., Sturgill, C. B. and Christiansen, K. D. (1999), *Designing A Successful HMD-Based Experience*, PRESENCE*,* 8, 469-473.

Quirk, A. D. (2003), *Navigational Systems and Desktop Environment Design Within the Virtual Space*, Proceedings of DCIT RHD Student Conference,

Quirk, A. D. (2005), *Navigational and Desktop Environment Design within Augmented and Virtual Reality Utilising Head Mounted Displays*, Proceedings of QUIDConf, 39-41

Sakaguchi, H. and Reinert, A. (2001), *Final Fantasy: The Spirits Within,* Sakaguchi, H. and Sakakibara, M., Columbia Pictures

Semwal, S. K. (2001), *Wayfinding and Navigation in Haptic VEs*, Department of Computer Science, University of Colorado at Colorado Springs

Serra, L., Hern, N., Guan, C. G., Lee, E., Lee, Y. H., Yeo, T. T., Chan, C. and Kockro, R. A. (1999), *An Interface for Precise and Comfortable 3D Work With Volumetric Medical Datasets*, Proceedings of Medicine Meets Virtual Reality: 7, 328-334

Steinicke, F. and Hinrichsy, K. (2006), *Grab-and-Throw Metaphor: Adapting Desktop-based Interaction Paradigms to Virtual Reality*, Proceedings of IEEE Symposium on 3D User Interfaces 2006, 83-87

Stoakley, R., Conway, M. J. and Pausch, R. (1995), *Virtual Reality on a WIM: Interactive Worlds in Miniature*, Proceedings of SIGCHI '95, 265-272

Sutherland, I. (1968), *A head-mounted three dimensional display*, Proceedings of Fall Joint Computer Conference, 33, 757-764

Valve Software, (2005), *Half Life 2*, Valve Software

Veen, J. (2000), *The Art & Science of Web Design,* New Riders, Indianapolis.

Wachowski, A. and Wachowski, L. (1999), *The Matrix,* Wachowski, A. and Wachowski, L., Warner Bros.

Warren, H. L. (2002), *Auditory Cueing Effects on Human Performance with an Adaptive System,* Master of Science, North Carolina State University, Raleigh.

Wingrave, C. A. (2001), *Nuance-Oriented Interfaces in Virtual Environments,* Master of Science in Computer Science and Applications, Virginia Polytechnic Institute and State University, Blacksburg, Virginia.

Wloka, M. M. (1995), *Interacting with Virtual Reality*, In *Virtual Environments and Product Development Processes*, (Ed, J. Rix, S. H., and J. Teixeira), Chapman & Hall.