Design and Evaluation of Domain-Specific Interaction Techniques in the AEC Domain for Immersive Virtual Environments

Jian Chen

Dissertation submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
In
Computer Science and Applications

Doug A. Bowman
R. Bowen Loftin
Christopher L. North
Manuel A. Pérez-Quiñones
Walid Y. Thabet

October 25, 2006
Blacksburg, VA

Keywords: 3D interaction, domain-specificity, design theory, usability engineering, virtual environment, evaluation method

Design and Evaluation of Domain-Specific Interaction Techniques in the AEC Domain for Immersive Virtual Environments

Jian Chen

ABSTRACT

Immersive virtual environments (VEs) are broadly applicable to situations where a user can directly perceive and interact with three-dimensional (3D) virtual objects. Currently, successful interactive applications of VEs are limited. Some interactive applications in the AEC (architecture / engineering / construction) domain have not yet benefited from applying VEs. A review of prior work has suggested that 3D interaction has not reached a level that meets real-world task requirements. Most interaction techniques pay little attention to the application contexts. When designers assemble these techniques to develop an interactive system, the interfaces often have very simple and not highly useful UIs.

In this work, we describe a domain-specific design approach (DSD) that utilizes pervasive and accurate domain knowledge for interaction design. The purpose of this dissertation is to study the effects of domain knowledge on interaction design. The DSD approach uses a three-level interaction design framework to represents a continuous design space of interaction. The framework has generative power to suggest alternative interaction techniques. We choose the AEC domain as the subject of study. Cloning and object manipulation for massing study are the two example tasks to provide practical and empirical evidences for applying the DSD.

This dissertation presents several important results of the knowledge use in the DSD approach. First, the DSD approach provides a theoretical foundation for designing 3D interaction. Techniques produced using DSD result in more useful real-world applications, at least in the domain of AEC. Second, the three-level interaction design framework forms a continuum of design and expands our understanding of 3D interaction design to a level that addresses real-world use. Third, this research proposes an integrated system design approach that integrates DSD and the usability engineering process. Fourth, this work produces a large set of empirical results and observations that demonstrate the effectiveness of domain-knowledge use in designing interaction techniques and applications. Finally, we apply domain-specific interaction techniques to real world applications and create a fairly complex application with improved usefulness.

Acknowledgements

The author would like to acknowledge the following people for their help, advice and support:

- Her advisor, Dr. Doug A. Bowman
- The members of the dissertation committee: Dr. R. Bowen Loftin, Dr. Christopher L. North, Dr. Manuel A. Pérez-Quiñones, and Dr. Walid Thabet
- Dr. Mehdi Setareh, for his comments and collaboration on Virtual-SAP
- Dr. Frederick P. Brooks, Jr. for his comments on the design theory and the architectural applications
- Dr. Francis Quek and Leelian Quek
- Dr. Drew Kessler for his help with the SVE toolkit
- Chadwick Wingrave, John Lucas, Nicholas Polys, Beth Yost, Gregorio Convertino, Young-Sam Ryu, Pardha Pyla, Michael Narayan, Tao Ni, and Mara Silva for their collaborations on various projects
- The 3DI and 3DUI group members
- Participants of usability studies for their contributions to this work
- Her parents Bingquan Chen and Keling Wang, and her husband Meng Ling
- NSF, for funding support (NSF-IIS-0237412)

To
my grandparents

Guanyi Chen 陈贯一
Jingyi Cheng 程景怡

Table of Contents

List of Figures

List of Tables

CHAPTER 1: *Introduction*

This chapter introduces the problems encountered while designing three-dimensional (3D) interfaces in the domain of architecture, engineering, and construction (AEC). A domain-specific design approach (DSD) is envisioned to improve the usefulness of interaction techniques. This chapter also describes the research questions and hypotheses, summarizes the contributions of this research, and finally outlines the organization of this dissertation.

1.1 Problem Statement

Immersive virtual environments (VEs) are synthetic, spatial (usually 3D) worlds seen from a first-person point of view. The view in a VE is under the real-time control of the user (Bowman et al. 2004). This vision, initiated in the 1960s (Sutherland 1965), has driven the human-computer interface in the decades since. Great strides have been made in advancing display technologies, increasing visual realism, reducing latency, and understanding perceptual and cognitive issues. Many researchers claimed that users gained better performance with VEs compared to conventional approaches (Kalawsky 1998).

Despite these advances, the number of practical VE applications remains small. Brooks reported only five categories of applications in "production" use (Brooks 1999), some of which are represented by only one or two examples. In the seven years that have passed since then, almost no new categories have been added. In contrast, in the field of interaction research, 1995 was a boom year for novel interaction techniques. But the boom slowed to a trickle with most recent publications making only slight improvements. Those applications used on a routine basis remain low in interactivity. The questions are why this is the case and where the divide is between interaction techniques and the applications.

To answer these questions, this research examines the domain of AEC to identify the problems. VEs are believed to be able to provide important benefits to the AEC industry (Brooks 1986, Durlach and Mavor 1995, Brooks 1999). For example, architects, contractors or engineers can use VEs to create, view, analyze, and modify their designs. A single VE can integrate all the functionality offered by other alternative technologies, such as desktop computer-aided design (CAD) tools, physical drawings, and physical 3D models. VEs also afford a first-person perspective and immerse users in the design space. In such environments, architects can verify their design and engineers can view simulation results (Kuester and Hutchinson 2005, Setareh et al. 2005).

In response to these needs, many VE prototypes for AEC have been developed (Szekely et al. 1993, Liang and Green 1994, Ullmer and Ishii 1997, Vries and Achten 2002). But the feedback (Bowman 1996, Bowman et al. 2003) we received from architects and engineers about using VEs as *design tools* had been mixed. They liked the ability to view the models at different scales while immersed, as well as the interactivity and flexibility offered by the system. However, they found that most applications had limited usefulness in supporting complex real-world design tasks. Modeling large, complex buildings that contain many beams and columns was a tedious task, because of numerous actions the user had to perform (Chen et al. 2004). Even if they could successfully construct the structure, frequent access to a three-quarter view was not easy using current interaction techniques.

As we have noted, tasks in the AEC domain are often complex and include flexible modification, changing viewpoints and so forth. 3D interaction has not reached a level to allow users to effectively perform such highly interactive tasks. Therefore, the lack of new categories of application could well be due to the lack of effective 3D interaction techniques. An analysis of current design practice of interaction techniques and user interfaces (UIs) in VEs further demonstrates this lack of effective interaction techniques. The following five observations are drawn based on the authors' experiences in designing 3D UIs and on a review of the literature.

- Interactive tasks are generic.
- Interaction techniques are generic.
- The conventional usability engineering (UE) approach applied to VEs does not increase the *usefulness* of 3D applications.
- Little is known about meeting domain constraints in designing interaction techniques.
- The design of 3D interaction and UIs are more complex than the design of 2D UIs.

First, interaction tasks are generic. Most well-studied tasks in VEs are the "universal tasks" of travel, selection, manipulation, system control, and symbolic input (Bowman et al. 2004). The idea is that these "universal tasks" are generic enough to form the fundamental building blocks of 3D user interfaces (UIs); many real-world tasks are a composition of these tasks. Therefore, if 3D interaction techniques are designed for these generic tasks, we can decompose high-level real-world tasks into these generic ones; then compose the corresponding generic interaction techniques to form the high-level techniques (Figure 1.1).

This decomposition is a powerful approach to simplify the problem-solving process by solving many small problems (Simon 1981, Rasmussen et al. 1994). However, the attempt to study the parts (here the generic interaction) in order to understand the whole (here high-level real-world interaction) is a reductionist or analytic view. "The whole is more than the sum of the parts" (Sanderson 2003). Often, important context information or relations between task conditions could be missed (Rasmussen and Vicente 1990). The resulting action sequence may not match users' understanding of how the work should be done, or may not lead to efficiency for the users (Vessey 1991).

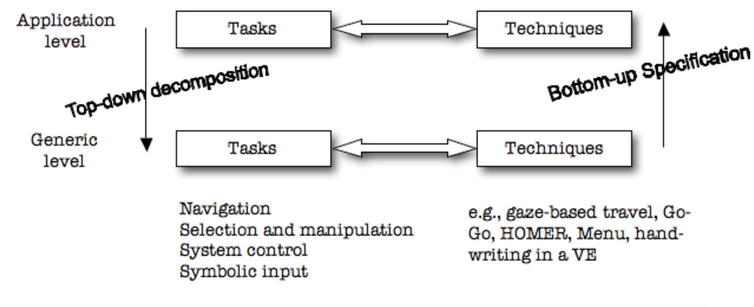


Figure 1.1. Conventional 3D Interaction Design Approach

Second, interaction techniques are generic. Most interaction techniques are designed for the afore-mentioned generic tasks. Since they are generic, they should have broad applicability. For example, Go-Go (Poupyrev et al. 1996) is a powerful object manipulation technique that can be used in any task conditions that require object manipulation. The technique allows the users to grab and place an object at a remote location without physically traveling there.

This generality, however, also limits the real world use of the generic-level techniques because real-world interaction always has constraints. For example, if the user's goal is to select an occluded object, the selection ray may need to be curved to reach the occlude objects (Olwal and Feiner 2003). If the user's goal is to place the object at an exact location quickly, the interaction should avoid the tracker or hand jitters by slowing down the movement of the virtual hand while the user's physical hand slows down (Frees and Kessler 2005). In summary, the interaction techniques designed should be the faithful representations of the tasks to be performed in real-world conditions.

Also because of the generality of techniques and tasks, most applications based on these interaction techniques do not have sophisticated user interfaces. Interaction techniques used in the most successful applications (Brooks 1999) are walkthrough or flythrough, i.e., simple travel from one location to another. Complex modeling systems beyond creating elementary shapes still do not exist.

Third, the conventional usability engineering (UE) approach (Hix and Hartson 1993, Hix et al. 1999), when applied to VEs, does not increase the *usefulness* of 3D applications. UE is a top-down approach (with iteration) focused on engineering the whole interactive system. The key is not in finding a *best* but instead a *working* design through experimental testing and iterative improvement (Wolf et al. 1989, Hornbak and Frokjar 2005, Wolf et al. 2006). This approach can work well for the desktop 2D UIs because standard windows, icon, mouse, and pointer (WIMP) user interfaces exist. However, there is no such standard in 3D UI design. Designers often need to step back to examine other design options to produce novel interaction techniques (Gabbard et al. 2003, Chen et al. 2004). Otherwise, VEs could be labeled as not suitable for those applications (Kaber and Riley 2001).

Fourth, Little is known about meeting domain requirement in designing interaction techniques. Such knowledge has been successfully applied in the area of whole-UI design (Sutcliffe et al. 1996), computer graphics, and vision (Marshall et al. 1980, Sloan et al. 2001, Wonka et al. 2003). For example, procedure modeling (Marshall et al. 1980, Wonka et al. 2003) utilizes semantic knowledge of components relationship to construct arbitrary model of a city or a single house. A rule as simple as "roof ~~ comp()" can split the roof into two pieces automatically to generate a slope (Muller et al. 2003, Muller et al. 2006).

Similarly in the human-computer interaction (HCI) discipline, Nardi (Nardi 1996) has advocated utilizing contextual knowledge in activity theory. Users' goal and work domain define a set of constraints within which a system should satisfy (Burns et al. 2001). The use of context information, a type of domain knowledge, has not been considered in the design of interaction techniques.

There are sparse techniques designed with domain constraints (Stuerzlinger and Smith 2002, Oh et al. 2006), However, the true effects of using domain knowledge in the design of 3D interaction techniques are still unknown. There is no framework or method for how to design useful and reusable 3D interaction techniques.

Finally, the design of 3D interaction and UIs are more complex than the design of 2D UIs (Stanney 1995, Kalawsky 1998, Sutcliffe 2003, Bowman et al. 2004, Burns et al. 2005). Designers need to make many design decisions, from choosing hardware devices and system development tools, to considering human factor issues (Stanney 1995, Szalavari and Gervautz 1997, Kalawsky 1999). Most designers are exhausted while implementing the system, let alone the development of new interaction techniques. There are also limited tools to support 3D interface design (Kessler et al. 2000, Kelso et al. 2002). Designers may have to do the work from scratch when designing interaction techniques.

These problems illustrate a clear divide between the real-world requirements for 3D interaction techniques and the available techniques that have been designed. This divide prompts us to explore a rigorous design approach to support the design of interaction for real-world use. We call this method the *domain-specific design* (DSD) method. DSD emphasizes domain knowledge use and reuse for creating 3D interaction techniques. Along the lines of previous research by Bowman (1999) for design at the generic level, DSD focuses at the domain level and emphasizes domain knowledge use and reuse. The ultimate goal is to increase the usefulness of the interaction techniques to meet real-world application needs. In this dissertation, a three-level framework is identified to help us understand interaction design and suggest many means to produce interaction techniques.

1.2 Definitions

Definitions of some of the key terms follow:

- **Virtual Environment (VE) / Virtual Reality (VR)**: A synthetic, spatial (usually 3D) world seen from a first-person point of view. The view in a VE is under the real-time control of the user (Bowman et al. 2004).

- **Human-Computer Interaction (HCI)**: A discipline concerned with the design, evaluation, and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them. HCI has two components: the interaction and the interface components (SIGCHI 1992).

- **Interaction**: Any communication between a user and a computer, be it direct or indirect in a context (Dix et al. 1993).

- **Interface**: A means that instantiates the interaction component (Dix et al. 1993).

- **Three-Dimensional (3D) Human-Computer Interaction**: Human-computer interaction in which the user's tasks are performed directly in a 3D spatial context. 3D HCI has two components: 3D interaction and 3D interface.

- **Domain**: A section of industry where requirements for computer systems will be found, or classes of design problems that pose particular challenges, e.g., graphics design, modeling or engineering, medicine, or visual analytics (Dix et al. 1993, Sutcliffe 2002). A domain can have sub-domains. For example, AEC is a domain that has three sub-domains: architecture, engineering, and construction. Architecture can be further divided into sub-domains including interior architecture, landscape architecture, and general architecture. A domain is bounded by a set of processes, data, objects, and artifacts that behave according to rules in that domain.

- **Domain Knowledge**: A composition of objects organized in class hierarchies, with attributes describing objects and relations. Domain knowledge structures complement task knowledge as actions change objects; domain knowledge also includes the user's spatial and episodic memory, as well as more abstract knowledge of the domain in categories and semantic networks (Johnson 1992, Sutcliffe et al. 1996). Domain knowledge affords reuse and usefulness (Sutcliffe 2002).

- **Three-Level Interaction Design Framework**: A framework that classifies interaction according to domain specificity. It has three levels: generic, domain-specific, and application-specific levels. The three levels form a continuum that all interaction techniques or tasks fall into the design space.

- **Generic Task**: A task consisting of navigation, manipulation, selection, or system control. These tasks are the fundamental "building blocks" of 3D user interfaces.

- **Generic Interaction Technique**: An interaction technique that is designed without any consideration of domain or applications in mind.

- **Domain-Specific Task**: A task that is framed with domain knowledge in mind.

- **Domain-Specific Interaction Technique**: An interaction technique designed with domain knowledge in mind.

- **Application-Specific Task**: A task that users want to achieve in order to meet functional requirements and satisfy some constraints of a particular application. Examples are painting a human figure (Keefe et al. 2001) and modeling a car surface (Grossman et al. 2002).

- **Application-Specific Interaction Technique**: An interaction technique that can be applied to a specific application only.

- **Domain-Specific Design (DSD)**: An approach that emphasizes the use of domain knowledge in the design process to frame design tasks or to design interaction techniques. Improving interaction design at this domain level is the key idea in this dissertation.

- **Usability**: The extent to which a computer system enables users to achieve specified goals effectively and efficiently while promoting a feeling of satisfaction in a given context of use (ISO/IEC9241-14 1998). Usability includes two levels: *usable* and *useful*. "Usable" stands for *operational* level usability such as factors that affect if buttons are clickable and if appropriate feedback is provided. "Useful" sets higher *functional* level criteria to define if the implementation can meet work-related requirements. A useful interface should be usable; but a usable interface or interaction technique might not be very useful.

1.3 Acronyms

2D: Two Dimensions or Two-Dimensional

3D: Three Dimensions or Three-Dimensional

AEC: Architecture / Engineering / Construction

DM: Direct Manipulation

DSD: Domain-Specific Design

HCI: Human-Computer Interaction

UI: User Interface

VE: Virtual Environment

VR: Virtual Reality

WIMP: Windows, Icon, Mouse, Pointer

1.4 Research questions

The overarching question the research detailed in the following chapters attempts to answer is:

> *What are the effects of designing domain-specific interaction techniques (as opposed to generic interaction techniques) with regard to usefulness, i.e., to accomplish work-related tasks?*

To attain this goal, this question has been divided into four sub-questions:

- *How is it possible to make sense of the design of 3D interaction? What should be designed?*
- *Can domain characteristics of the AEC domain be used to design domain-specific interaction techniques?*
- *How can domain-specific interaction techniques be developed?*
- *Will the usefulness of 3D applications be improved by using domain-specific interaction techniques produced from DSD? How can usefulness be measured?*

1.5 Hypotheses

The following hypotheses have guided the research in this dissertation:

- *Compared to generic techniques, overall usability and usefulness will increase with the use of interaction techniques designed at the domain level with regard to efficiency, effectiveness, and match with task requirements.*
- *It is possible to make use of domain characteristics to design or redesign interaction techniques leading to greater usability.*
- *Applications developed with the addition of domain-specific interaction techniques will be more useful.*

1.6 Scope of this Research

There is seemingly a chasm between 3D applications and current interaction techniques. Most highly interactive systems need useful interaction techniques, but most interaction techniques are too generic to be applied. To bridge the chasm, the research in this dissertation focuses on *what* to design and *how* to design interaction techniques.

First, this dissertation focuses on the DSD method, because theory-guided design can have broader impact than providing technological improvement to address point problems. Existing design theories in the 3D UI literature have focused on a guideline-based approach, but there is little research on what and how to design for 3D interaction. The DSD method defines *what* needs to be done in order to apply the interaction effectively; the three-level design framework classifies interactive tasks and techniques and defines *how* techniques can be produced.

Second, this dissertation also focuses on how to produce useful interaction techniques to address real-world use. Domain knowledge serves as the key contribution factor to the DSD framework to produce domain-level interaction techniques. In this dissertation work, we study the effects of the domain knowledge use. We provide evidences that DSD is useful for producing interaction techniques though we do not evaluate DSD by comparing it to other design methods.

Third, the studies conducted in this work stress the usefulness of interaction techniques. This research distinguishes between *usable* and *useful*. An interaction technique or a user interface is said to be usable if it is operational. An interaction technique or a user interface is said to be useful if it can meet work-related task requirements. Therefore usefulness is at a higher level that emphasizes the effectiveness of usability. A system that is usable is not necessarily useful. However, a useful system has to be usable first. This dissertation addresses usefulness because the "usable" aspect of usability is well studied in the 3D interaction literature (Hix et al. 1999, Bowman et al. 2002, Stanney et al. 2003).

Fourth, the AEC domain is chosen as the subject of this research. Our work is rooted in Virtual-SAP (Bowman et al. 2003) and other architecture applications (Butterworth et al. 1992, Bowman 1996, Mine 1997, Bowman et al. 2003). We choose the domain of AEC because applications in this domain are often highly interactive. Also, researchers believe that VEs should benefit this domain, but it is not happening. The AEC domain probably has the longest history of using VEs since the first walkthrough application (Brooks 1986). A large number of UI designs for AEC in immersive VEs exist (Sachs et al. 1991, Bowman 1996, Mine 1997, Shaw and Green 1997, Regenbrecht et al. 2000, Schkolne et al. 2001). Still, there are not VE applications that architects use on a daily basis.

From the experimental point of view, choosing the AEC domain can also avoid confounding factors that could invalidate the empirical studies we conduct. This is because many years of experience of designing desktop tools have provided a well-defined task space. We can derive domain knowledge from those 2D desktop tools to study the effects of domain-specific design. For example, the cloning task used in our study exists in AutoCAD, 3D studioMax, sketchUp etc.

Fifth, there are two phases in this dissertation work. The first phase is a trial-and-error exploration of the use of DSD to design interaction techniques for the cloning task (Chen et al. 2004). We tested the effects of domain knowledge use for interaction design. An integrated UE process reflects the specific design process we took. The second phase is to follow the integrated UE process to produce a useful user interface for massing study.

Finally, the focus is not on the details of the actual capture of domain knowledge, but instead on the general use of domain knowledge and its intrinsic demands in designing interaction techniques, regardless of how domain knowledge is obtained. This distinguishes our work from existing research in many knowledge-based or domain-oriented 3D or 2D systems (e.g., Feiner and MacIntyre 1993, Sutcliffe et al. 1996, Fischer 2001).

1.7 Contributions

This dissertation research makes a number of contributions to the field of designing 3D interaction, VEs, and HCI.

- Our understanding of 3D interaction techniques is extended from generic 3D UI design and evaluation to domain-specific design and evaluation.

  The work described in this dissertation has investigated the DSD approach: the method, the framework, it's usefulness, and it's applications. We have proposed the use of domain knowledge in the design of interaction techniques. This approach might be among the first to try to make interaction techniques more suitable for real-world use in VEs.

- Our research offers a three-level design framework that provides an instrument for producing interaction techniques.

  Once DSD has been recognized for showing *what* to design, the three-level design framework is similar to a template for showing *how* to design. Designers can take advantage of adding domain knowledge about the current design task by testing alternative paths in the framework. Using the framework and the DSD method, many domain-level techniques can be produced. The framework is the first to make sense of current design and can be used as a generative framework for the design of domain-level techniques.

- This research provides design theory that has increased the usefulness of real-world applications. If applied to other disciplines, DSD has the potential to increase the use of VEs.

  After integrating techniques produced from the DSD approach, those techniques lead immediately to more useful applications compared to the previous systems built with generic-level interaction techniques.

- This research provides a novel empirical evaluation approach migrated from social science to address the evaluation of usefulness of applications.

We did not compare this approach with other usability evaluation approaches (such as heuristic evaluation, formative, and summative evaluations) in this dissertation work since the evaluation method is not the focus.

- This work produces many empirical results and interaction techniques that can be used by practitioners. Several toolkits were constructed in the course of this research that can potentially simplify the future development of 3D UIs and interaction techniques.

  During the course of this study, many direct and indirect manipulation techniques based on the pen-and-tablet metaphor interface and tabletop interaction were designed, iteratively improved, and empirically evaluated. Guidelines gathered for their use can be applied in general to improve the design of 3D UIs.

- This work also provides an integrated usability engineering process that reflects the process used in Virtual-SAP. We produced the design process for two reasons. First, domain-level design will not be useful if it cannot increase the usefulness of applications. Second, Virtual-SAP might be one of the most complex and useful 3D UIs for architectural design. The design process can help other designers follow the same process to improve upon existing designs or produce new ones.

1.8 Organization of this dissertation

This dissertation consists of eight chapters. This chapter, *Introduction*, has introduced the subject of DSD, motivated by the need for designing more useful applications through the domain-specific design of interaction techniques. It also provided a description of research objectives, questions, scope, and hypotheses.

Chapter 2, *Related work*, presents a detailed look at previous work that has influenced this research and provided background for the current research. Areas covered include the concept of 3D interaction, current state-of-the-art interaction techniques, and design theories.

Chapter 3, *Domain-Specific Design Theory*, presents the design methodology for creating interaction techniques, with all of its components discussed in detail. It provides a description of what to design and how the three-level design framework can help the development of domain-specific interaction techniques.

Chapter 4, *Designing Domain-Specific Interaction Techniques for Cloning*, describes two sets of interaction techniques for cloning and an initial empirical study. One set of interaction techniques are widget-based UIs. The empirical study demonstrates that we can utilize domain characteristics to design domain-level interaction techniques. The other set of interaction techniques represents the first case study of applying the DSD method. All interaction techniques are mapped to the three-level design framework.

Chapter 5, *Evaluating the Effects of Domain-Specific Interaction Techniques*, describes two comprehensive empirical evaluations that study the effects of domain knowledge use on interaction techniques and on the Virtual-SAP application accordingly.

Chapter 6, *Reflection on the System Design Process*, describes the system development process that integrates DSD and usability engineering. The potential benefits of this integrated approach are summarized.

Chapter 7, *Designing a Domain-Specific 3D User Interface for Massing Study*, presents the second case study using DSD in the design of a 3D user interface for massing study. It serves as an example of applying the DSD method and the integrated usability engineering process to design 3D interaction techniques and a 3D user interface.

Chapter 8, *Conclusion and Future Work*, summarizes the major results and draws conclusions from them. It highlights the main contributions of this research and makes suggestions for future work in this area.

CHAPTER 2: *Related Work*

This chapter presents a detailed look at previous work that has influenced our research or provided background for the current research. Areas covered include: three-dimensional human-computer interaction, interaction and interface, domain knowledge for interactive system design, and design theory.

2.1 Three-Dimensional Human-Computer Interaction Concepts

2.1.1 Definition of 3D Human-Computer Interaction

Three-dimensional human-computer interaction is defined as "human-computer interaction in which the user's tasks are performed directly in a 3D spatial context" (Bowman et al. 2003). SIGCHI defines *Human-computer interaction* as a discipline that concerned with the design, evaluation, and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them (SIGCHI 1992). This definition places all phenomena relevant to 3D human-computer interaction under one umbrella. Since HCI is multi-disciplinary in nature, these phenomena can include perception and motor skill, cognition, hardware devices and ergonomics, software tools, and interaction techniques. Interaction and interface are the two components of HCI (Dix et al. 1993). This dissertation focuses on interaction; and the goal is to improve the usefulness of the 3D interface through effective interaction techniques.

2.1.2 Definition of 3D Interaction

3D Interaction means any communication between a user and a computer, be it direct or indirect, in a 3D context (Dix et al. 1993, Hix and Hartson 1993). The aim is to help users accomplish their tasks and to increase productivity from some application domains. Interaction design is often described according to tasks (what to design for) and techniques (how to accomplish the tasks). For example, a task of moving a sofa from a corner of a room to another is a manipulation task, and the Go-Go interaction technique (Poupyrev et al. 1996) can help the user to accomplish this task.

Since 3D interaction occurs in a 3D spatial context, the mapping between human actions and the corresponding actions in computer environments is used to distinguish 3D interaction from 2D interaction. Therefore, a 3D world that presents spatial information does not have to involve 3D interaction. For example, a medical doctor dragging a slider widget to control the transparency of a virtual visible human body is not using a 3D interaction technique (Figure 2.1). In the same application, if the widget is displayed in the space around a human body and users control a widget in that 3D spatial context, then a 3D interaction has occurred (Figure 2.2) (Lin et al. 2002).

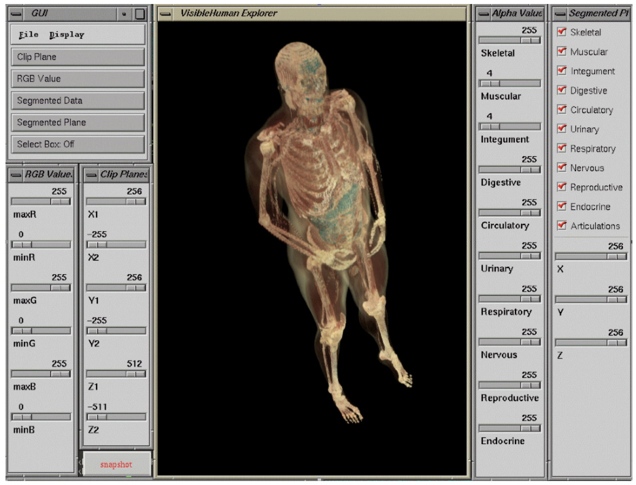


Figure 2.1. Users can drag sliders to control transparency, the clipping plane, or the display of the segmentation information.

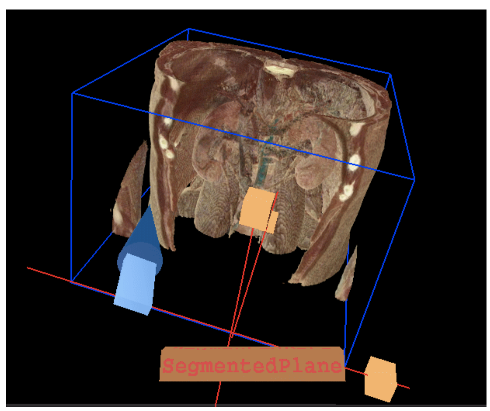


Figure 2.2. Users use spatial widgets to specify the cutting planes on a ImmersaDesk®

3D interaction techniques can also be used to control 2D data, such as dragging a geographic map or opening a folder. For example, if the user uses a stick attached with Vicon® trackers to pan and zoom a map, then 3D interaction has occurred. So 3D interaction *occurs in* an UI. UI can include many components, such as displays, platforms, information presentation mechanisms, input or output devices, or widgets (see Section 2.1.3).

2.1.3 Definition of 3D User Interface

The second component of 3D human-computer interaction is the 3D "interface", i.e., any user interface that involves 3D interaction or 3D graphical displays. The study of the 3D interface has included many dimensions, such as real-time rendering (Fuchs et al. 1985, Pizer et al. 1986), hardware devices (MacKenzie et al. 1991, Zhai and Senders 1997), software tools (Kessler et al. 2000, Kelso et al. 2002), and the mediation of the interaction between human and computers in a 3D setting. For example, the physical dimension includes input and output devices (Hinckley et al. 1994); the perceptual and cognitive dimension includes users' problem solving skills and cognitive ability (Green 1989, Leganchuk et al. 2000, Lok et al. 2003, Sutcliffe et al. 2006); and motor behavior and ergonomic studies serve the goal of users' effective interaction with the system that provides optimum comfort (MacKenzie and Ware 1993, Zhai et al. 1999, Zhai et al. 2003).

In this dissertation work, we do not address any of these dimensions alone. We rather focus on the process and theory of design, i.e., how to design 3D UIs to improve the usefulness of the target applications. But all of these dimensions undoubtedly affect the design of interaction techniques and 3D interfaces; therefore, they have to be considered when designing techniques.

2.1.4 Importance of 3D Interaction

We live in a 3D world and much of the work all of us do in daily life is spatial in nature. Since 3D interaction allows users to work in 3D, it should be suitable for many spatial tasks. For example, Holt and his co-authors (2004) found task efficiency doubled in a VE for a pipe-routing tasks compared to the conventional physical full-size physical prototype approach. VEs are also more cost-effective because they do not need the re-construction of scaled or full-sized physical prototypes. In a VE, users can choose the types of cables and route them as many times as they wish before the real system is built. Other VEs share the same benefits, such as training systems (Loftin and Kenney 1995) and modeling environments (Grossman et al. 2003). Astronauts and medical doctors can be re-trained without imposing extra cost.

3D interaction has not only been applied to VEs. Most nascent technologies, such as augmented reality (AR), ubiquitous computing (UbiComp), and large-screen displays, may all gain benefits from using 3D interaction. For example, in a ubiquitous environment, a user can gather information about a printed file by placing his/her hand next to a printer (Feiner and MacIntyre 1993). In an outdoor AR application, the user can pinch his/her fingers to arrange virtual models on an image plane (Piekarski and Thomas 2003). While using a large-screen display in a shared environment, the location of the user's body can become an input to implicitly control the content of the information. The information can be displayed at a fine detail when the user is close by or at an overview level when s/he is further away (Vogel and Balakrishnan 2004).

3D interaction also provides rich and compelling experiences through multimodal input and haptic feedback. Biologists can understand molecular structures better when viewing in 3D. Force feedback can be sensed by users when they perform molecular docking tasks (Brooks et al. 1990). Multimodal interactions, such as gesture or voice input, are intuitive and create a "put-that-there" experience while immersed (Billinghurst 1998, LaViola et al. 2001).

Given all these advantages of using VEs, we should see a boom of VE applications. This is not happening. When we carefully examine the seven applications at the production stage reported by Dr. Frederick Brooks (Brooks 1999), we find that five use the advantages of VEs in constructing expensive simulators to replicate reality and to display visually appealing graphics. Interaction with these interfaces was also designed to replicate reality. Molecular docking using the nanomanipulator (Brooks et al. 1990), though rich in interaction, is another example of replicating reality. The architectural layout in the same paper reported only design review. The major benefits lie in the collaborative viewing of data but not the interactivity.

Designing 3D interaction techniques for VEs is difficult, unfortunately. The physical world is full of constraints. A laptop will remain on a table when we place it because of gravity and collision. In VEs, such constraints often do not exist. The lack of constraints in VEs does not necessarily reduce users' performance but it increases the complexity of design for designers. Designers have unbounded freedom to design anything they like. There could be many bad designs.

The limitation of hardware devices and human cognitive and motor abilities also limit 3D interaction techniques. For example, a stylus would be more suitable for pen-input UIs compared to a wand, which would be more suitable for pointing and selection in 3D. The environment itself can also affect the effectiveness of 3D interaction techniques. For example, interaction techniques would be quite different if the environment was highly dense. 3D environments do not often provide rich information spaces that would help users gain a better understanding of the world. There are also ergonomic issues such as cumbersome output devices (e.g., head-mounted display), jittering trackers, and cybersickness. A user's standing posture also prohibits the use of VEs in a daily basis. A designer has to consider all of these factors while designing a system.

2.2 Interaction Techniques

2.2.1 Classification by domain specificity

The term "domain-specificity" is used to convey how much the designed techniques or the tasks are related to the real-world use. The most common design approach is an intentionally "broad brush" so that interaction is generic enough to be applied to every application. Those universal tasks of navigation, selection, manipulation, and system control belong to this level. For example, HOMER (Bowman and Hodges 1997), Go-Go (Poupyrev et al. 1996), and World-in-Miniature (WIM) (Pausch et al. 1995, Stoakley et al. 1995) are generic interaction techniques. Using HOMER, a virtual hand in the world can "shoot" a ray to select an object. The object is then attached to the end of the ray and moved by reeling it in or out with a joystick, or moved by a mapping function, which could be Go-Go scaling. HOMER is very intuitive and can almost be used in any environment that requires object manipulation.

There can be no doubting the value of this approach; because designers have to understand these low-level designs before proceeding to more complex real-world design tasks (Bowman et al. 1999).

An alternative approach to designing interaction techniques is to directly address real-world functional tasks in the system design process. These are tasks explicitly stated by the end users and those implicitly defined by the specific application. The functional tasks often impose constraints on the design. Therefore, this type of 3D interface is generally highly useful for the specific target application. For example, one neurosurgical visualization tool used a doll's head and a Plexiglas plate so that a surgeon could define a cutting plane (Hinckley et al. 1994). CavePainting uses props coupled with gestures to control the color and the width of the stroke (Keefe et al. 2001). One CAD system used a physical tape to draw splines or other types of curves in 3D (Balakrishnan et al. 1999). However, one drawback of these highly specific systems is that it can be difficult to apply these 3D interaction techniques in other contexts.

2.2.2 Classification by design styles

Although the domain-specificity classification helps us understand interaction, it does not suggest what approaches designers can take to create interaction techniques. Examining design style can suggest design ideas adopted by good designers. These include: the **metaphor-based approach**, **constraints**, **using proprioceptive cues**, and **multimodal interaction**. These styles have been effective means to create effective interaction techniques.

Real-world metaphors are often used because they are intuitive. For example, a virtual mirror or magnifying glass allows the users to see objects behind them or to examine a model at a different level of detail (Gleicher and Witkin 1992, Kopper et al. 2006). But not every natural interaction is efficient. The virtual-hand technique (Robinett and Rolland 1992, Vince 1995) forces the user to grab an object directly by "touch". The object moves at a 1:1 scale to the user's hand movement. Imagine grabbing an object that is 10 meters away; the user would have to move to that location first before performing the main manipulation task.

Some magic interaction techniques work more efficiently. For example, HOMER (Bowman and Hodges 1997) and Go-Go (Poupyrev et al. 1996) scale up arm movements so objects can be placed at a distance as well as nearby. The user can use ray-based selection coupled with landmark-based travel to perform the manipulation task of grabbing an object. The user can point to the target object using a ray to indicate the selection, and then place the object at the desired location.

The pen-and-tablet metaphor user interface (Angus and Sowizral 1995) is a well-known 2D WIMP style UI adapted to 3D. This metaphor makes use of a pen that has six degrees-of-freedom (DOFs), and a tracked tablet. Users can issue a command on a widget-based UI (Bowman et al. 1998, Lindeman et al. 1999) or draw strokes to express richer information based on handwriting and gesture recognition (Poupyrev et al. 1998). Such a metaphor can also be used for path planning for navigation and sketching.

Besides metaphor-based UIs, constraints are often added to avoid the disadvantages of high degree-of-freedom operations or to exaggerate users' motor, perceptual, and cognitive abilities. A constraint-based system implicitly makes assumptions to aid user's intention according to user behavior. For example, a hand movement slowing suddenly could mean that the user is likely to place an object (Frees and Kessler 2005). A multiple view system enables the user to choose the location of a camera to see different parts of the world without moving to that location. A distorted view provides a "focus plus context" information space (McGuffin et al. 2003). A user can also select and move objects from multiple views (Lucas et al. 2005).

Semantic constraints or physics-based constraints have been used to specify objects' behaviors. A virtual teapot can be intelligent enough to sit on top of a table when the table is repositioned or reoriented. The advantage is that users can predict certain behaviors according to their properties in the physical world (Bukowski and Sequin 1995). Objects can also behave according to a particular domain of use (Kitamura et al. 1998, Goesele and Stuerzlinger 1999, Kitamura et al. 2002, Rosson and Carroll 2002, Wonka et al. 2003). For example, a wheel can only rotate around it's axle after installation.

Proprioceptive or depth cues can increase the effectiveness of interaction. For example, over-the-shoulder deletion allows the user to delete objects effectively by throwing objects behind himself (Mine et al. 1997). WIM allows fast manipulation or travel based on a small world held in the user's hand (Stoakley et al. 1995).

Input styles greatly affect interaction techniques. 3D interaction techniques may use head/hand motion, gaze, manipulation of physical objects, speech, gesture, or posture; some use the combination of two or more. Given we have so many interaction techniques designed, still they are not applied to real-world applications.

2.3 Domain Knowledge Use in Interface Design

Since all tasks are situated in some context, a task in a particular application must meet some conditions to leverage applicability. Such conditions, as defined in knowledge engineering, can be task features, assumptions, suitability criteria, resources and process characteristics, and method ontology. If such conditions are from a specific domain, they form domain knowledge.

Domain knowledge has been used as a source of design to produce effective interfaces (Sutcliffe et al. 1996). Much user interface research focuses on theoretical frameworks for modeling knowledge as affordances and constraints and the goals of the user as states of the domain. For example, Gulliksen and Sandblad (1995) and Sutcliffe (2003) rework Norman's conceptual model of HCI to include UI components. The idea is to provide cognitive resources, design features, and associated requirements to designers so that appropriate design decisions can be made. If applied, the system image (what has been built), the design model (concepts held by designer), and the user's model (user's mental model) of the task and actions should be a perfect match so that the two gulfs (of evaluation and execution) are bridged.

Knowledge representation has also been integrated into software engineering tools to build systems that can recode design rationale, support reasoning, and create artifact memories (Maiden and Sutcliffe 1992, Fischer 1993, Sutcliffe et al. 1996).

But domain-knowledge has not been applied to design interaction techniques except a few, such as constraints-based interaction (Bukowski and Sequin 1995, Oh and Stuerzlinger 2005). Most domain knowledge is used for task modeling (Forsyth and Ponce 2002), rather than interaction design in immersive VEs.

2.4 Design Theory

3D interaction lacks an accepted design theory. Since the focus of this dissertation is on the design, we review the most influential design theories in the literature. They are originated from Simon (1981), Schon (1983), and Cross (2000) that stress different aspects of design.

Simon's design theory (1981) treats all design problem as "wicked", i.e., what the problem is and what the solution should be are dependent on each other; the problem and the solution co-evolve. The process of finding solutions to design problems is divided into two processes. The analysis step breaks down activities so that problems could become small solvable problems. The synthesis step is a 'building up' process, i.e., to integrate the solutions for each small problem to form the overall solution space.

The decomposition approach is a successful means of problem solving, especially when dealing with large complex problems. An example of this process in HCI is hierarchical task analysis (HTA) where tasks and subtasks are placed in boxes and links that specify the activities. With decomposition, tasks are easier for the users to understand and for the designers to implement. This is because most user interface entities, objects, and even user's intentions have this property of being built of hierarchical structures.

Other approaches to perform thorough task analyses in HCI include interviewing, reading notebooks, holding training sessions using prototypes or scenarios, or using case-based approaches (Button 2000, Rosson and Carroll 2002, Visser 2006). In the 3D UI design, both task analyses and HTA can be applied.

Simon's theory also emphasizes the generation of many design solutions and then lets the use sort them out. This suggests that we should create a framework that would allow designers to create many design options. In our work, the three-level design framework serves this goal.

Cross's engineering approach (2000) is similar to Simon's decomposition and composition. But Cross's approach prescribes the design process by analyzing how designers think. In this way, other designers can follow the prescribed process to design a good system. One example of such a process is to use a decision tree. Unlike HTA, which is pure task decomposition, the decision trees also link boxes to indicate the action flow.

All possible solutions to each node and their rationale are indicated based on specific design requirements. Similar approaches used in HCI include rationale analyses and claims (Carroll 2003) and instrumental interaction (Beaudouin-Lafon 2000).

The third approach to design is called reflection-in-action (Schon 1983). Schon emphasizes the talk between the designer and the system. This process constitutes the language of design. The meta-language allows reflection on the action of design. For example, an architect uses sketches and a spatial-action language to represent concepts thus enabling experimentation. The talk and experiment are iterative. The designer then evaluates her/his design every step of the conversation. This reflection theory is applicable to many language tools to generate notations and to evaluate the interface.

2.5 Summary

This chapter has reviewed interaction techniques, design approaches, and theories that have motivated or affected this work. It has claimed that interaction is an important factor to bring effective 3D applications (Brooks 1999), but that existing interaction techniques and design approaches for 3D interaction have not been able to meet real-world needs.

CHAPTER 3: *Domain-Specific Design Theory*

This chapter describes DSD, the theoretical foundation of this dissertation. As we have reviewed in Chapter 2, most interaction is designed at either a generic level or an application-specific level. In this research, we propose to design at a domain level sitting in the middle ground. The rationale is to use domain knowledge to guide interaction design for reuse and usefulness. Cloning techniques are used as an example of DSD.

3.1 Three-Level Design Framework

From our experience with the Virtual-SAP application (Bowman et al. 2003) and our examination of existing 3D UIs, we have recognized that interaction design can occur on three levels. The classification is based on domain specificity, i.e., how much the designed techniques or the tasks are related to the use of domain knowledge. The three levels are the generic, domain-specific, and application-specific levels, listed in order of increased specificity (Figure 3.1). In this framework, we follow conventions by describing interaction design according to tasks (what to design for) and techniques (how to accomplish the tasks). In the following sections, we first describe the two existing levels of design at the application and the generic level, and then describe in detail the design at the domain level.

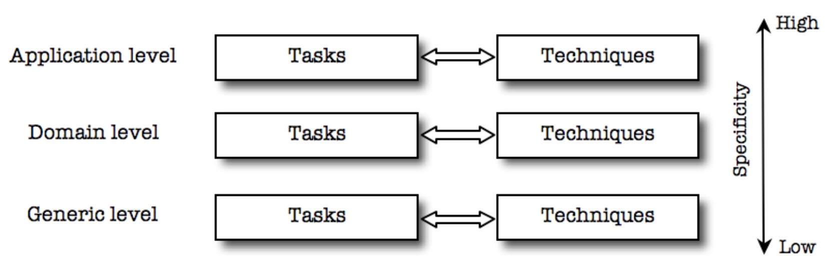


Figure 3.1. Three-level design framework for 3D interaction techniques

3.1.1 Generic Level

The generic-level tasks include travel, selection, manipulation, system control, and symbolic input (Mine 1995, Poupyrev et al. 1997, LaViola et al. 2001, Ruddle et al. 2002, Bowman et al. 2004). They are commonly observed in a wide spectrum of problems across various applications. For example, the applications of modeling a building, psychiatric treatment, gaming, and visualizing a bat flying are completely different applications. But they all involve travel tasks, i.e., moving users from place to place.

Applying travel techniques (such as point to fly, joystick navigation, or orbit viewing) can accomplish this task.

A technique is called a generic-level technique if it is designed without considering any specific application context. Design at the generic level has the following properties.

- Well-defined design space: There are at least two types of taxonomies that describe the design space of tasks or techniques (Bowman et al. 2004, Dachselt and Hubner 2007). The first approach is to define taxonomy according to task sequence. For example, a travel technique can be described in three steps: find the destination, then plan a route, and finally travel there. The second approach is to specify parameters that could potentially affect interaction. For example, the user's arm length, the distance of travel etc (Poupyrev et al. 1997, Bowman and Hodges 1999).

- Widely available evaluation results: Testbed design and evaluation is effective to measure task performance and user's behavior (Poupyrev et al. 1998, Bowman et al. 1999, Tan et al. 2001).

- Low-level universal tasks: Tasks are at a low level that contains direct references to one or several actions performed on objects (Johnson 1992). Typically, high-level tasks can be decomposed to these generic tasks but not any further. While designing an overall UI, designers simply integrate well-known techniques.

- No context of use: Techniques are designed without any consideration of context of use. For example, Go-Go (Poupyrev et al. 1996) is a generic interaction technique because it can be used in any environment that requires travel or manipulation.

- Lowest in specificity/highest in reuse: Because generic tasks and techniques are designed without considering the context of use, they are not specific. Therefore, the techniques are reusable across applications.

- Tweaking for real-world use: Designers often need to adapt generic interaction techniques to meet real-world requirements.

3.1.2 Application Level

An application-level task is a task at a *functional* level framed for a particular application without any abstraction. A technique is called an application-specific interaction technique if a specific application is considered. For example, connecting two pipes in the above-ceiling application is an application-specific task. Techniques designed for training simulators (Brooks 1999) would be application-specific techniques. This is because the interaction techniques vary for different training purposes and the training simulators. Design at the application level has the following properties.

- Less well-defined design space: Because the high-level tasks vary from application to application. There could be almost no overlap between tasks. Interface designers may have to design very specific interaction technique for that task given specific application requirements.

- Designed with context and activity in mind: the tasks are framed and the techniques are designed within a specific application context. Some applications have the set of functions within which activities take place (Vicente 1999).

- Highly specific/less reusable: Because interaction techniques are designed within a specific application context, they should be highly useful and usable in that context. However, reusing the interaction techniques in any other context could be limited. For example, the interaction technique used in one neurosurgical visualization tool uses a doll's head and a Plexiglas plate so that a surgeon could define a cutting plane (Hinckley et al. 1994).

3.1.3 Domain Level

Ultimately, 3D UIs have to be application-specific to some degree to ensure usability and usefulness for real-world work. But the design of 3D interaction techniques does not always need to take place at the application level. This is because we can abstract *functions* into *constraints* when a pattern of artifacts or other common characteristics and knowledge are shared among different applications. For example, building a high-rise 50 level building is a functional task; whilst cloning is a task that abstracted from the requirement. The cloning does not exist in real world but is an abstract representation of task that could enhance the user's job performance. We call the design at the domain level domain-specific design (DSD).

Domain-specific tasks are generalized from the application level that can be gathered through task analysis or modeling process (Johnson 1992). Domain-specific techniques are produced with applications' contexts that can be reused when many applications share common characteristics. In this way, we may also reduce the designers' effort in creating 3D UIs because techniques are applicable to the application directly. Our claim is that the use of domain-specific design results in higher usability and usefulness as compared to design at the generic level. Domain-specific design also allows greater reuse as compared to design at the application-specific level because of knowledge is reusable.

3.1.3.1 Domain

A *domain* is a section in industry where requirements for computer systems will be found or classes of design problem that pose particularly interesting challenges (Sutcliffe 2002). A domain can have sub-domains. A domain is bounded by having it's own processes, data, objects, and artifacts that behave according to rules that are set by that domain, therefore it is possible that a study of a domain, or domain analysis, can determine the generalized characteristics of a particular domain. In a practical sense from our point of view, a domain is an area to which we can apply the technologies or techniques of VEs or 3D UIs; thus we sometimes use the term "application domain".

3.1.3.2 Domain Knowledge

Domain knowledge helps us frame design tasks and design interaction techniques. It is defined following the task knowledge structure (TKS) model (Johnson 1992):

> *"Domain knowledge is composed of objects organized in class hierarchies, with attributes describing objects and relations. Domain knowledge structures complement task knowledge as actions change objects; however, domain knowledge will also include the user's spatial and episodic memory, as well as more abstract knowledge of the domain in categories and semantic networks."*

Both semantic knowledge and spatial knowledge have been applied to modeling and object creation in previous work. For example, the instancing texture method uses semantic knowledge of the architectural domain to model large structures (Wonka et al. 2003). The model is generated using a stochastic, parametric L-system. A set of rules controls the transformation, scaling, extrusion, and branching of the building geometry. Procedural modeling combines the topological relationship or constraints of the different model parts to generate new buildings (Parish and Muller 2001). An example of the topology could be that windows have to be on a wall. The topological knowledge is also applicable to object manipulation, e.g., a lamp on a table is moved with the table. There is also work that supports spatial knowledge, in which objects follow the laws of physics. For example, objects do not float in space but sit on top of each other. These constraints have improved users' task efficiency (Bukowski and Sequin 1995, Wonka et al. 2003, Oh and Stuerzlinger 2005).

Artifacts are a type of knowledge that represents the context of work. Such knowledge can also guide 3D UI design. Carroll (Carroll and Rosson 1992, Rosson and Carroll 2002) proposes the scenario-based design to address the users' context. Activity theory emphasizes the use of artifact-mediated HCI (Nardi 1996).

In our work, domain knowledge is used to guide the design of interaction techniques for several reasons.

- Stability: Domain knowledge is relatively stable compared to the application. Because the design is an ongoing process of interpretation and reshaping, designers benefit from having a stable collection of domain knowledge (and tasks and techniques based on it) even the system is constantly changing (Sutcliffe et al. 1996, Chen and Bowman 2006).

- Intuitiveness: Domain knowledge is generally intuitive or systematic for users. An interface designed according to that knowledge typically provides a better mental model and understanding of the environment (Carroll and Olson 1988). Users' familiarity with the task definition often makes the interface more intelligible and predictable. For example, object cloning techniques in the architectural domain is easy to understand because most buildings have symmetrical shapes and repetitive patterns. These patterns are accepted by architects and are taken as design bases.

- Feasibility: Domain knowledge can be collected effectively by knowledge elicitation techniques (Wielinga et al. 1992, Gordon et al. 1993, Gordon 1994), so the domain-specific design of interaction techniques is feasible.

- Reusability: Knowledge is shared among applications; therefore techniques can be reused within a domain or even in another domain with similar characteristics.

3.1.3.3 Domain-Specific Tasks and Techniques

A domain-specific task is an abstraction of domain activities that can be shared among many applications. For example, the cloning task used in this dissertation is a domain task since it can be used for all modeling applications. Techniques designed with domain characteristics in mind are called domain-specific techniques.

Activities define what users usually do in the real world, while goals and sub-goals of an activity are described as tasks through task analysis (Moran 1983, Diaper 1989, Kirwan and Ainsworth 1992). An activity can be realized by techniques designed for those tasks and sub-tasks coupled with action sequences. For example, modeling a large structure is an activity that can be realized through cloning operations.

So when do we reach domain-specific implementation? An interaction design style, such as widgets, proprioceptive cues, metaphor-based interaction, or gestures can never be domain specific in themselves but design styles. We cannot call these design styles domain specific until we add semantics or domain knowledge to them. For example, sliders are not domain specific, but the sliders used for cloning is a domain specific way to realize the cloning task because the design reflects the use of domain constraints of repetition, regular shapes etc. Additionally, the domain characteristics are not obvious until one tries to actually specify the design space within certain applications.

3.2 Domain-Specific Design of Interaction Techniques

Domain-specific design is relevant to the shared knowledge among applications. There exist many methods to conduct knowledge elicitation (Wielinga et al. 1992, Gordon et al. 1993, Gordon 1994) which is also an active research topic by itself. We ask questions: how can we produce domain-specific interaction techniques assuming we can derive common knowledge from different applications? What are effects of design at this domain level compared to at a generic level regarding domain specificity?

Figure 3.2 shows the possibilities within our three-level framework for designing domain-specific interaction techniques. The numbers next to the arrows are used for the purpose of discussion. To differentiate traditional approaches from our DSD approach, we use the annotations "()" and "<>" to represent traditional approaches and DSD respectively.

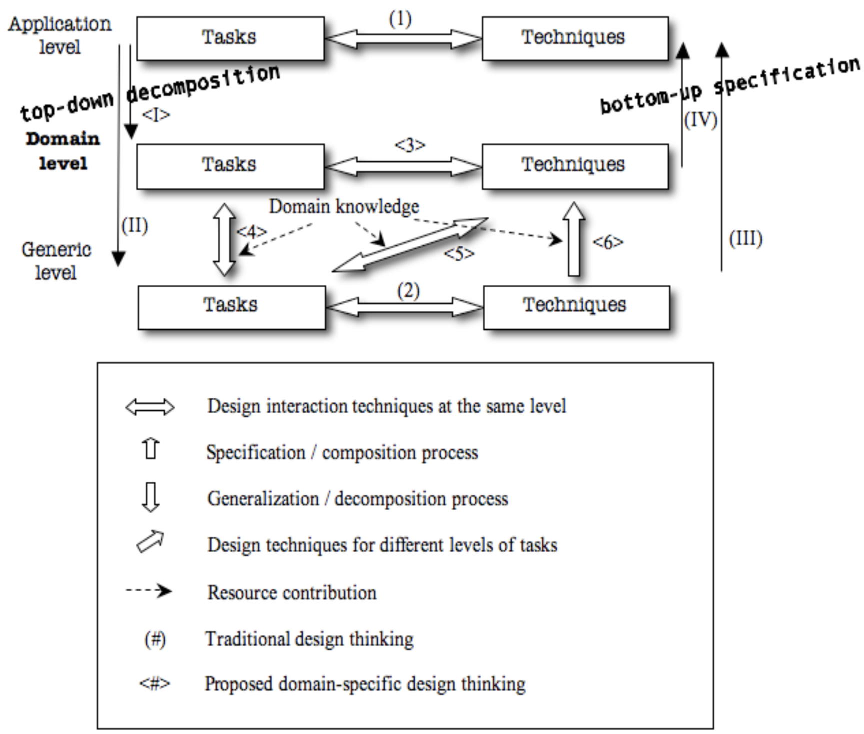


Figure 3.2. Three-level design framework with the emphasis on designing domain-level interaction techniques

3.2.1 Formation of Design Tasks

User tasks can be gathered either following path <I> or path (II) though task analysis (Gould and Lewis 1983, Kirwan and Ainsworth 1992, Sutcliffe 2002). Path <I> is the generalization of an application-level task to a domain-level task, while path (II) represents the use of generic-level tasks (universal tasks) as a substitute for an application-level task. Several task analysis methods have been proposed (Diaper 1989, Kirwan and Ainsworth 1992). One way is to develop the mapping from the user's conceptual model of the application to tasks through analogy (Moran 1983).

3.2.2 Mixed Approach to Design Interaction Techniques

The conventional method of designing 3D interaction techniques follows a mixed approach: top-down decomposition (path (4)) followed by bottom up specification (path <5> or (2)+<6>). For example, cloning is a domain-level task. It was decomposed to copy-and-paste then further decomposed to selection and manipulation; the generic Go-Go technique was then chosen (path (2)) and tweaked to work for the domain by snapping to grid (path <6>).

3.2.3 Bottom-up Approach from the Generic-Level to the Domain-Level

Another way to design domain-specific interaction techniques is by specialization, i.e., design starts from generic-level techniques, which are transformed into domain-level techniques by the addition of domain knowledge (path <6>).

For example, HOMER (Bowman and Hodges 1997) is a common 3D manipulation technique at the generic level. The copy-by-example technique (Figure 3) improves HOMER by utilizing the repetitive characteristic of the objects in the AEC domain. Instead of users placing each object individually, copy-by-example only requires the user to place the first copy, then the position of the next is automatically calculated when the user issues a "paste" command through a button click. Using this technique, the user still has the flexibility of moving the copies on the fly and the program will update the next position accordingly.

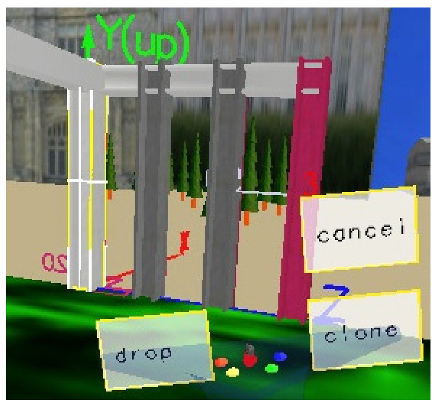


Figure 3.3. Copy-by-example interface for cloning (Copy-by-example)

Domain-specific interaction techniques can also be designed for a generic-level task (path <5>). In this process, generic level interaction techniques are not necessarily taken into account. Domain requirements can impose specific needs on generic tasks; the according domain characteristics can empower interaction techniques. For example, object selection is a generic-level task. Since related objects in the AEC domain are often near to one another and laid out on a rectangular grid, a 3D selection box can be used to select multiple objects of interest (Lucas et al. 2005). The selection box is not a refinement of an existing generic selection technique; it is a domain-specific technique designed directly for a generic object selection task.

3.2.4 Horizontal Path at the Domain Level

We can design techniques directly for a task at any level using the horizontal paths in Figure 3.2. Traditionally, this has been done at the generic level (path (2)) or application level (path (1)), as we noted in section 2. Of course, we can also do this at the domain level (path <3>). For example, once we identified cloning as a domain-specific task, we were able to design many different techniques specifically for the cloning task (see Chapter 4 for more details on any of these techniques).

The conventional way to do cloning in desktop tools is by numerical input. Therefore, we designed a keypad interface (Figure 3.4) for this purpose. The slider widget technique (Figure 3.5) is another example of a domain-specific technique designed for cloning specifically.

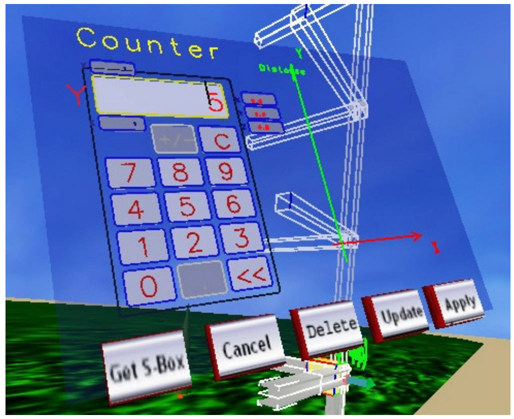


Figure 3.4. Keypad user interface for cloning (Keypad)

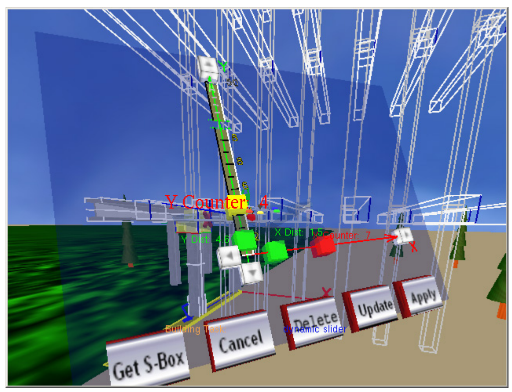


Figure 3.5. Dynamic slider user interface for cloning (D-slider)

PORT is a 3D resizing technique (Lucas et al. 2005). The users can resize an object along the axis to which they are pointing in 3D space. Since we were already using PORT to resize the 3D selection box in our architectural environment, we tweaked PORT to work for cloning to maintain consistency. Using PORT (Figure 3.6), users can specify the number of copies and the distance between copies using the same combination of pointing and joystick input as the resizing technique.

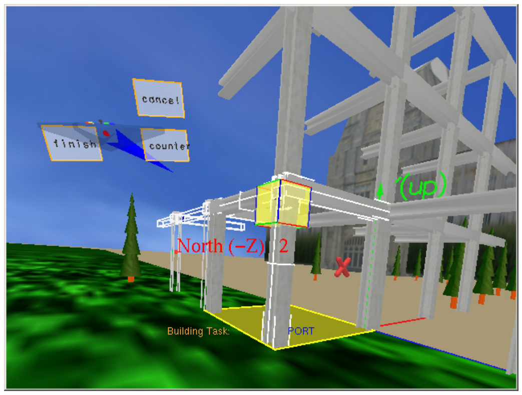


Figure 3.6. PORT for cloning (PORT-C)

Another cloning technique is called the space-filling metaphor interface (Figure 3.7). This technique allows the user to explicitly define the space that a building will occupy. The actions are the same as PORT, but the number of copies and the distance between them are determined by the system once the space is fixed, i.e., increasing the distance between copies may decrease the number of copies.

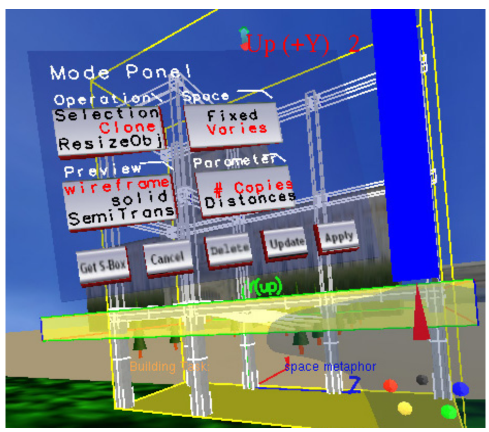


Figure 3.7. Space-filling metaphor user interface for cloning (Space-filling)

Designing interaction techniques that directly address tasks might seem very intuitive at first glance. Paradoxically, this rarely happens at the domain level. UI designers seem to be constrained by reusing low-level techniques. This seems especially true for nascent technologies (Beaudouin-Lafon 2000).

3.2.5 Taxonomy-based Testbed Approach

In designing these techniques using path <3>, we first developed a taxonomy for the domain-specific task of cloning (Figure 4.2, Section 4.2). We then designed techniques by considering different ways to specify each of the parameters in the design space. The keypad, slider, PORT, and space-metaphor techniques are all designed to allow the user to specify the number of clones, the distance between clones, and the direction in which the clones will be created, but the user's actions are quite different in each technique. Keypads and sliders are conventional ways of specifying numeric parameters, while

PORT and the space-metaphor technique use direct manipulation with spatial input. They cover the interaction design style of direct and indirect manipulation.

3.2.6 Comparison of Different Paths

In a sense, the three-level framework carries the properties of being descriptive, comparative and generative (Beaudouin-Lafon 2000). It is descriptive because it incorporates both existing (e.g., generic level and application-level design) and new (e.g., domain level) design approaches. It is comparative because the paths between levels suggest a way to compare alternative designs. Finally, it is generative because it can facilitate the creation of new interaction techniques by suggesting new ways of design thinking.

3.3 Strengths of the Domain-Specific Design Approach

It is useful to summarize some of the strengths of the domain-specific design approach. The strengths serve as the design rationale for DSD. In some cases, however, our points are potential benefits or goals that may not always be present that requires empirical validation. Nonetheless, when compared with the pure generic or application-specific design approaches, the integration of domain-specific interaction techniques in the framework offers a number of benefits.

- DSD increases ”context” in the designed techniques; and therefore increases the usefulness of resulting applications that adopt domain-specific interaction techniques.

Because we engage the users and advocate collecting domain knowledge at the beginning stage, the tasks and techniques reflect users’ thoughts, intentions, and work activities. For example, we used scenarios to describe the problem definition and actions that could possibly be implemented by the system (Rosson and Carroll 2002, Chen et al. 2004). We used taxonomies to define a task space so that techniques can directly address those domain needs (Johnson 1992, Shneiderman 1998). We have shown that domain knowledge affects the task formation and the design of interaction techniques. Also, because the contextual information is better used, the resulting application will have a higher level of usefulness (see Chapter 5 for empirical evidence).

- DSD is cost-effective.

If every 3D UI design must start with the generic level of interaction techniques, or worse starts from scratch, this would not be cost effective. Because the design is often required be done in a short period of time, we can shorten the development stage if we have a good collection of domain-specific interaction techniques. Additionally, the domain-specific concepts and knowledge grow as the design moves from domain to domain (Sutcliffe et al. 1996). Within specific areas of work, designers can refine and develop the concepts appropriate to the new domain. Therefore, we will never start from scratch but instead build upon our growing knowledge and techniques. Also because we carry domain knowledge that can be shared within many applications, there is a possibility that the techniques will be reusable. Having achieved design rationale and scenarios for an artifact, it may be possible to generalize the findings to the artifact genre, rather than narrowly to the specific artifact (Carroll 2000).

- DSD can simplify the integration process.

Because design is necessarily a process of hierarchical decomposition and composition, the techniques designed for a certain task granularity have to be re-examined after they are composed together to determine if the higher-level technique evaluates well (Hix and Hartson 1993). In DSD, the decomposition stops at the domain level, so there is no need to compose basic interaction techniques. Designers will also have fewer conflicts to resolve during the integration process. For example, in our case, cloning is not further decomposed (i.e. no copy and paste action for cloning is needed). The PORT interaction technique is applicable to resizing, cloning, and object selection. The reuse has led to greater consistency of the UI.

- DSD produces different design options that can be better utilized by the UE process.

UE can discover usability problems with existing designs but does not support the generation of different design options. DSD can create many design alternatives by following different paths with the context information and domain knowledge integrated. These two processes can be combined to design effective 3D UIs.

- DSD can enrich the generic-level design space.

The use of DSD can lead to the recognition of new tasks, and thus can enhance the design space at the generic level. For example, object or scene modeling can be considered as a generic-level task because it is needed in many applications. Cloning, for which we have designed many interaction techniques, is a type of modeling sub-task, and may be useful even outside the AEC domain.

- DSD encourages a new type of design thinking.

The three-level framework in DSD opens up a large number of design possibilities that would not be apparent to many 3D UI designers, as shown by the variety of paths in Figure 3.2. It makes clear the distinction between tasks and techniques, and offers a middle ground between the extremes of application-specific and generic design approaches. For example, as we have found in the design of object creation techniques for AEC, sometimes we need to design for a new type of task, as opposed to designing new techniques for the universal tasks.

3.4 Summary

We have classified the interaction to three levels: generic, domain-specific, and application specific levels based on how much the interaction techniques produced is specific to an application.

We call the "universal" tasks of navigation, selection, manipulation, and system control generic interaction tasks because they are considered as the fundamental building blocks of all interactive systems. Almost all other interactive tasks can be represented by these generic tasks. Interaction techniques that designed without any consideration of applications in mind are called *generic level interaction techniques*.

On the other end of the continuum is the *application level design*. It meets *functional* requirements of an application. For example, painting with a phantom device, drawing a surface of a car body, and viewing three cutting planes of a virtual model are application level task. Interaction techniques designed with such specific functionality in mind are called application specific interaction techniques.

Domain knowledge use defines *domain-level tasks and techniques*. Domain knowledge often shares among many applications. For example, such knowledge can be the properties of the artifact (lamp on the desk), the process of making the artifact from its description (move the lamp while moving the desk), the explicit knowledge (e.g., the beam has to have certain thickness in order to sustain a certain force) or a parameter space associated with the constraints (e.g., numbers of copies, space are part of cloning taxonomy), etc.

It might be hard to draw a hard line between *constraints* and *functions*; however, it is useful to distinguish these two because functions are the primary reason that the interaction techniques or certain UI system is desired. Design knowledge is better to be used to generate the candidate designs based on constraints so interaction techniques can be reusable.

CHAPTER 4: *Designing Domain-Specific Interaction Techniques for Cloning*

This chapter presents the completed portion of interaction techniques for cloning. The techniques are divided into two sets marked different phases of this study. In phase one, we intent to learn the possibilities of using domain characteristics to design domain specific interaction techniques and validate the design space for cloning. We describe the design and evaluation of our novel interaction techniques with emphasis on the design rationale and design style.

In phase two, we design interaction techniques that are mapped to the three-level design framework and the DSD approach to examine the use of DSD and tradeoffs of different path. The empirically evaluation of the use can be found in Chapter 5.

4.1 Problem Scenario

We address the following problem scenario to design cloning techniques.

> *Lisa, a city planner, plans to build residential houses in a crowded city. She is planning the layout using immersive virtual environment (VE) techniques. She can walk through the model and move houses around using the current state-of-the-art VE interaction techniques, for example, Go-Go for objects manipulation and travel…. She has modeled a house. This is fairly easy to do. She also wants to duplicate the house since all houses are the same. This process is tedious however. She has to create a copy one by one by following an action sequence of selection, copy-then-paste, confirmation, and then repeat this sequence again and again. Rather, she would go back to AutoCAD on her desktop, duplicate the house, and reload the model into the virtual world…. Once Lisa loads the large world, she finds that Go-Go travel is not good enough for helping her move quickly to access a three quarter view…*

Apparently, the 3D interface is able to enhance user's perception in understanding the environments and the design by providing a one-to-one scale and immersion. However, the UI has not been able to accommodate and augment the power of the interaction to meet Lisa's needs for easily creating many houses and examining the outcome. We solve Lisa's problem by designing cloning techniques (Figure 4.1).

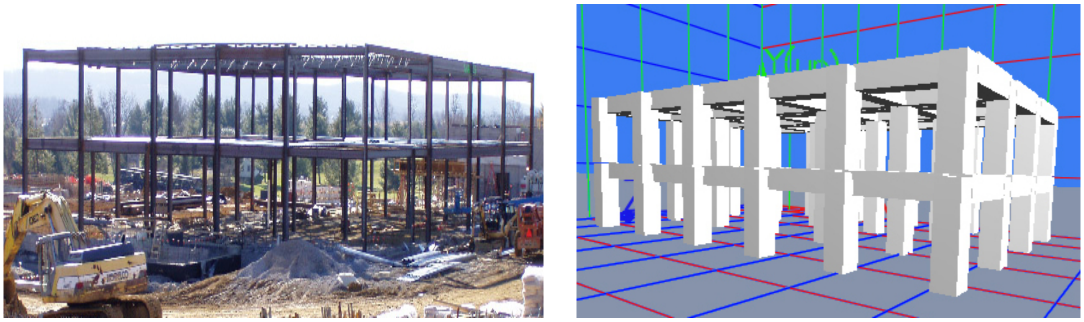


Figure 4.1. Example structure at a real world construction site and in our environment

4.2 *Cloning Task*

Cloning is defined as the process to generate multiple spatially distributed copies of an object to form a repeated pattern. It has been widely used on desktop tools such as AutoCAD, 3D Max, SketchUp® and others. We choose cloning as an example to examine the possibility to conduct domain specific design and the effects of domain specificity.

Cloning tasks have two major components: *selection* and *clone generation*. The user first *selects* objects of interest in the virtual world, then specifies the properties of the cloned objects. In this study, we fix the object selection technique and only consider the cloning step in order to validate the experiments. All object selection is performed using PORT interaction techniques (Lucas et al. 2005).

We use taxonomy to describe a set of parameters for cloning describing how the newly generated objects are distributed in space and their visual attributes (Figure 4.2). Taxonomy is effective in integrate parameters that are associated with a specific task in order to design the interaction techniques (Poupyrev et al. 1998, Bowman et al. 1999, Chen et al. 2004). The taxonomy can be used to design a testbed to measure the effectiveness of interaction techniques. Guidelines of use are also written so that techniques can be easily chosen by practitioners that wish to apply the techniques to other applications.

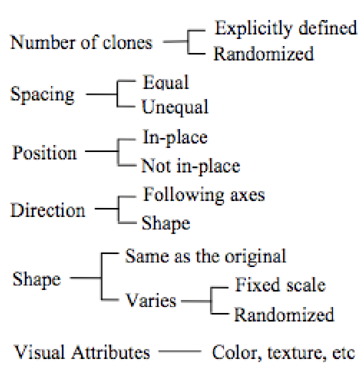


Figure 4.2. Design space for cloning techniques

The *number of clones* can be a number explicitly defined by the user or randomly generated by the system. *Spacing* defines how far two adjacent copies are from each other, which could be equally or unequally distributed in the space. *Position* includes in-place, where the newly generated structure is part of the original structure, and not in-place, allowing the new structure to be put in any other position. *Direction* defines the area in which the clones are laid out, which may follow the axes of a coordinate system, or may follow a certain shape (e.g., a function defined by several points selected by the user). *Shape* and *visual attributes* deal with the look and feel of the objects.

For example, in order to construct a building shown in Figure 4.1, the user can request 10x10 copies along the horizontal plane (e.g., along the x and z axes) and one copy along the vertical axis; she could then adjust the space between the houses to 25 feet.

This set of parameters for cloning focuses mainly on qualitative parameters and not on semantic constraints or rule-based attributes. For example, assume an environment with 10 tables, with a teapot on one of the tables, and suppose a user wants to put a teapot on each of the tables. Such tasks are not addressed in our parameter space.

4.3 System Setup and Selection Technique

Interaction techniques for cloning are designed using the HMD based system. The display device used in this study is a Virtual Research V8 head-mounted display (HMD) (binocular, 640x480 resolution and 60^o diagonal field of view). The user's head and hands are tracked with an InterSense IS-900 VET tracking system. A pen-and-tablet interface (Angus and Sowizral 1995) includes a wand and a piece of tracked Plexiglas (Figure 4.3). There are two different sizes of the tablet. The sizes are 25cm x 25cm (Figure 4.3, left) and 48cm x 35cm (Figure 4.3, right) accordingly.

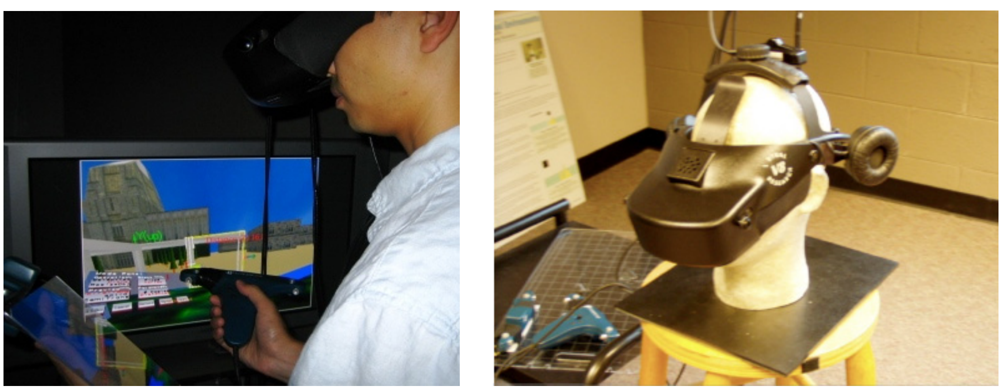


Figure 4.3. Physical devices used in this study

The virtual world is based on the campus of our university. Participants have a good mental map and would not get disoriented. In other words, wayfinding was not a confounding factor while we conduct usability studies.

The travel technique is pointing based (Bowman et al. 2004), i.e., users point in the desired direction with the wand and hold a button (upper-left) to fly. A multiple object selection technique, PORT (Lucas et al. 2005), is integrated into the system in this application. Users select multiple objects with a selection box. Any object(s) intersected with this box are selected and highlighted. Selection and deselection activate sound cues to provide extra feedback.

The selection box can be grabbed with ray-casting (Mine 1995), manipulated with HOMER (Bowman and Hodges 1997) and resized with PORT (Lucas et al. 2005) (Figure 4.4). PORT uses the relative direction of the hand-held pointer to the object to determine the axis of resizing. Pushing the joystick forward/backward resizes the side to which the ray is pointing. An arrow and an X marker indicate the direction of resizing. The box is always aligned to the principal axes of the world coordinate system to reduce the degrees-of-freedom a user has to control. This choice is made because most structural elements are aligned to the axes.

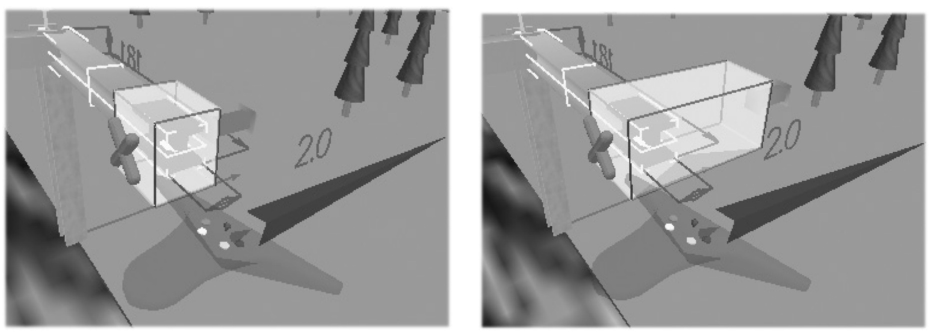


Figure 4.4. Pointer orientation-based resizing technique (PORT)

4.4 *Indirect Cloning Techniques (Set 1)*

The first set of cloning techniques includes widget-based slider UIs. They are the numerical spinners interface (Spin), the orthogonal 1-DOF sliders interface (1-DOF-O), the cavalier 1-DOF sliders interface (1-DOF-C), the 2-DOF widgets interface (2-DOF), and the 3-DOF widgets interface (3-DOF). Each of these interfaces allows the user to quickly generate complex, repetitive models. We designed many interaction techniques also to learn how widgets UIs function as they can be potentially used in many applications.

4.4.1 Numerical spinners interface (Spin)

The *numerical spinners interface* (Spin) directly displays the nine variables on the tablet (Figure 4.5). Spinners were used to control the input of the number of clones and the spacing parameters; and radio buttons were used to change the direction, either on the positive or negative side of an axis. The display was organized in a clearly readable manner: three rows for the x, y, and z axes and three columns, for the parameters "counter" (for number of clones), "dir" (for direction), and "dist" (for spacing).

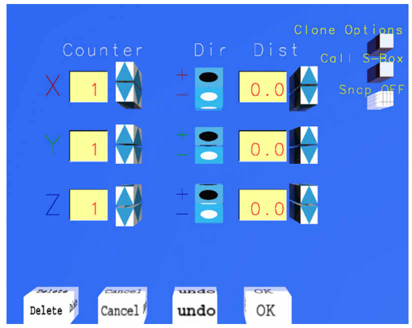


Figure 4.5. Numerical spinner interface (Spin)

Users clicked the up and down arrows to increase or decrease the values. The corresponding variables were displayed in the text fields with a yellow background. The color black denoted the toggling radio buttons' current state. We also color-coded the text displays to match the axes drawn in the world (red for x, green for y, and blue for z).

4.4.2 Orthogonal 1-DOF sliders (1-DOF-O)

Slider widgets are often used in user interfaces to define variables. An example in 3D user interfaces is the work of Chen (1988), who grouped the sliders together on the interface or attached them to objects within the world. Users manipulated the variables by directly controlling the widgets in 3D space.

We used six sliders to control the nine variables, two widgets on each axis to control the counter and distance variables. The interface was called *orthogonal 1-DOF-sliders* (1-DOF-O), because all sliders were constrained to move along one dimension (Figure 16). They were displayed in different colors: red-colored sliders controlled the number of clones and green-colored sliders controlled spacing. The direction variables were automatically accounted for by the spacing widgets since we allowed these widgets to indicate both positive and negative values. We constrained the movement of the "number of clones" widgets to be along the positive axes because these variables are inherently positive.

To avoid clutter on the tablet, only four widgets were displayed at a time. The two widgets on the x axis were always visible. Either the two widgets on the y axis or on the z axis were visible depending on the angle between the tablet and the horizontal axis. If the angle was within a range of 0^{o} to 45^{o} (Figure 4.6 (a)), the z axis and the two attached widgets were displayed; and if the angle was within a range of 45^{o} to 90^{o}, the y axis and its widgets were displayed. We chose to use angle to determine the mode because it was fast and easy for the users to rotate the tablet while performing the task. They only need to make a small adjustment of their hand or arm's position to switch axes. Such visibility constraints were made inactive when the user was interacting with a widget. This was because the user might get confused if the widget s/he was interacting with suddenly became invisible.

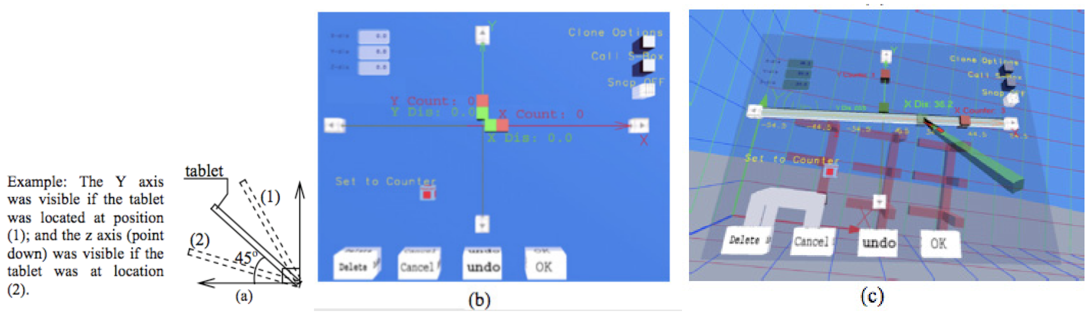


Figure 4.6. Orthogonal 1-DOF slider interface (1-DOF-O)

The interface also included *fine-adjustment widgets*. Clicking the arrow buttons shown at the end of each axis would move either the number of clones or spacing widgets along one dimension with predefined increments. Selection of which widget to move was controlled by another widget shown on the left bottom side of the tablet. "Set to counter" indicates that clicking on arrows would move counter widgets, and toggling it to "Set to distance" would cause the movement of spacing widgets on the tablet. The behavior of the arrow widgets was consistent on all interfaces.

The annotation and color themes used on the tablet were carefully designed for easy reading. The annotations were drawn on the screen (as a heads-up display). The current values of the variables were displayed next to the widgets in the same color as the widgets. The same color scheme was used to display the axes in the world. Also, the negative axis directions were drawn in a very different brown color for ease of interpretation.

4.4.3 Cavalier 1-DOF sliders interface (1-DOF-C)

Similar to the previous interface, in the cavalier 1-DOF sliders interface (1-DOF-C), all slider widgets were confined to move along one dimension. Instead of drawing the y and z axes along the same line on the tablet, we drew a 45^{o} cavalier projection in order to separate the y and z axes (Figure 4.7). All widgets were visible on the tablet.

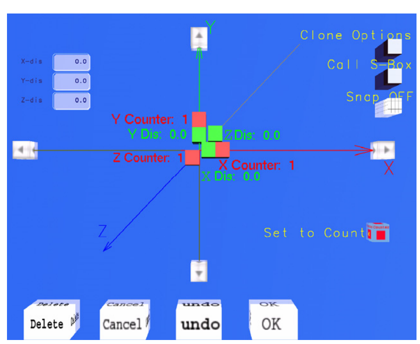


Figure 4.7. Cavalier 1-DOF sliders interface (1-DOF-C)

This projection visually gave the users a 3D feeling although it was a 2D user interface. We enhanced this effect by displaying the positive z axis larger than the negative z axis, which was visually further away. The behavior of all slider widgets was similar to those in the 1-DOF-O interface except that the z axis sliders were confined to move along the oblique z axis.

4.4.4 2-DOF widgets interface (2-DOF)

Widgets used in 2D desktop applications typically have few DOFs: they usually move along one dimension or along an axis. Sliders in the previous user interfaces were similar to their desktop counterpart. This is not a necessary constraint for widgets in 3D interfaces, where widgets can be moved within space.

We designed widgets whose movement was constrained to a plane, and therefore the interface was called the 2-DOF widgets interface (Figure 4.8). Dragging the widgets causes two variables along two directions to change simultaneously. For example, dragging the slider shown in Figure 4.8(b) could change the number of copies in both the x and z directions. Similar to the 1-DOF-O interface, subsets of widgets were visible based on the angle of the tablet. Therefore, only two widgets were on the tablet at any given time.

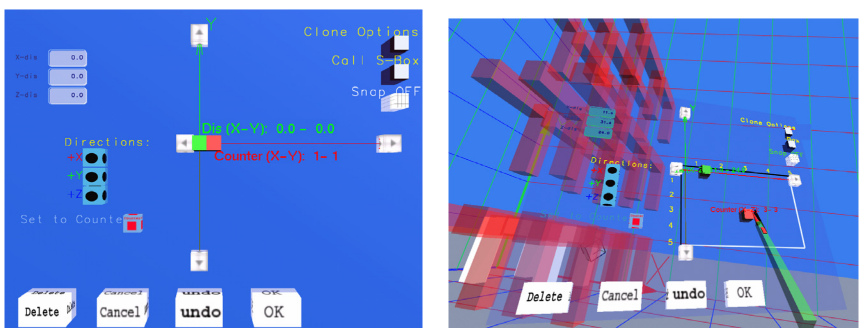


Figure 4.8. 2-DOF widget interface (2-DOF)

This interface provided for widgets' movement along the positive axes only. The direction was controlled by separate toggle widgets shown on the left side of the tablet. We chose this because the increased DOFs would make positioning tasks harder. In addition, the variables that were already set up were displayed next to the widgets.

4.4.5 3-DOF widgets interface (3-DOF)

We increased the DOFs even more in the 3-DOF widgets interface, by allowing the widgets to be dragged in three dimensions within a box area defined by the size of the tablet (Figure 4.9). The x and z axes were located on the tablet, and the y axis pointed out of the tablet. Only two slider widgets were needed for this interface: one controlled three "number of clones" variables and the other controlled three spacing variables (with directions). Because we wanted to put fine-control widgets at the end of each axis, the y-axis arrow widget ended up floating in the space above the tablet.

Since one objective of our study was to test how different design options affected usability, we combined the direction and the space variables again. A grid was drawn which looked similar to the one in the virtual world in order to assist the user to decide the direction. We added a toggle button marked "Change Y Dir" which allowed toggling the y direction on the tablet since the origin was located at the center of the tablet and the negative y direction was beneath the tablet, and therefore was difficult to reach.

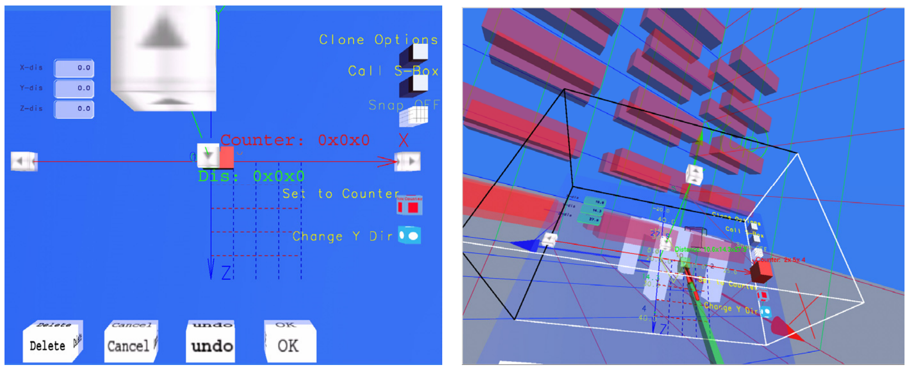


Figure 4.9. 3-DOF widget interface (3-DOF)

4.4.6 Design Considerations

We considered cloning as an interactive process: users repeatedly changed the position of widgets, visually evaluated the outcome, re-adjusted the widgets, and repeated this process until the desired cloning result was obtained. Such a process created a fast feedback loop between the user input and result that significantly speeded up the process of interpreting the interfaces, making an action plan, and executing the tasks (Norman 1986).

We also tried to learn how different designs affected users' attention. We believed that a better interface would allow users to concentrate on their primary task rather than on the interface. Finally, continuous response and feedback were provided because of the constant interaction between the user and the application.

In the remainder of this section, we describe the rationale for several design choices that we applied to most or all of our cloning interfaces.

4.4.6.1 Self-explained constraints

We designed widgets to provide a perceived affordance for spatial inputs, to convey the behavior, and to inform the user how to act on them. We drew *slider slots* for 1-DOF sliders, boxes for 2-DOF widgets, and a cube for 3-DOF widgets upon selection (Figure 4.6(c), 4.8(b), 4.9(b)). We did not make them visible in the initial user interface due to limited space on the tablet and the overlap of the distance and counter axes. Visual clutter and overlap would make the interface hard to interpret, therefore increasing the cognitive load.

4.4.6.2 3D Preview

The common visual feedback for all of our cloning interfaces was a *3D preview*, i.e., previewing the result as the user interacts with the tablet. This provided a direct *what you see is what you get* (WYSIWYG) interface. The red-colored semitransparent objects displayed in Figures 4.6(c), 4.8(b), and 4.9(b) is the preview results corresponding to the user's current input. This output allowed users to see if their actions were furthering their goals.

The preview made generating the result an effortless and natural process, and may minimize the "consciously calculated activity" (Badler et al. 1986). Also, users do not need mental rotation (Shepard and Metzler 1971) to visualize the results since the structure was displayed directly in the virtual world. This design feature bridges the gulf of evaluation (that is, it reduces the amount of effort required to interpret and understand the actions of the system) (Norman 1986), making results more easily interpretable.

4.4.6.3 Fine-control widgets

The fine-control widgets or arrow widgets allowed the movements of widgets to be constrained to a single dimension. 2-DOF or 3-DOF widgets might allow fast creation of large models, but the higher DOFs that users need to control may be frustrating because great agility and manual dexterity are required. Constraints, on the other hand, limit such movement to allow fast operation.

4.4.6.4 Appropriate annotation and visual feedback

Another source of feedback was semantic in nature (Green and Jacob 1990, Butterworth et al. 1992). Text was drawn on the tablet to indicate the current parameters in the system. The annotations around the widgets play two roles: (1) they help users choose the correct widget; and (2) they provide feedback for the values of variables. Furthermore, they were color-coded to augment the correspondence between the widgets and the text displayed.

4.4.6.5 Volume-based widgets

Fitts' law (Fitts 1954, Zhai et al. 2003) suggests that the target acquisition is affected by the size of the target. We used both *volume-based widgets* (widgets with a magnetic area around them) and visual feedback to provide ease of selection. The volume-based widgets made the interface less cluttered and afforded easy selection. Also, widgets were highlighted upon touch.

4.4.6.6 Integration of users' domain knowledge

In the 1-DOF-C interface, a cube was drawn to represent the overall size of the cloned object within the space. This additional visual feedback might increase the understanding of the cloned structure in an abstract manner, and the users do not need to travel to get an overview of the structure being cloned. We displayed this representation in this interface because of its natural structure and spatial relationship with the axes.

4.4.6.7 Integrality and separability

The cloning task is a multi-dimensional input task. Jacob and his co-authors (1994) raised an important issue about choosing the right input devices for tasks stating that performance improves when the perceptual structure of the task matches the control structure of the input device. If this is to be true, users who perceive the task structure as an integral pattern should like to act in 3D given the input device is 3D in nature. On the other hand, users who do not should have a city-block pattern while performing the task.

4.5 Evaluation the Feasibility and Design Space of Cloning

Cloning task is a domain task and we have defined the design space according to domain characteristics. We made efforts to design several cloning techniques to examine if the design space is reasonable. A good cloning technique is one that allows the users to accomplish tasks easily and efficiently with little or no discomfort. We performed an exploratory study to evaluate the usability of the cloning interfaces we had designed. The purpose of the evaluation was to (1) compare subjective usability responses aimed at investigating understandability, learnability, and ease of use, and (2) find usability problems from the perspectives of designers and other users regarding the content, aesthetics, and interaction techniques used in our system.

4.5.1 Participants

We recruited eight participants for this study. Two participants had construction domain knowledge and user interface design experience. The other six had user interface knowledge but no construction domain knowledge. They were all graduate students at Virginia Tech.

4.5.2 Tasks

Two tasks, a *matching* task and a *numeric* task, were used in the experiment. For the *matching task*, a miniature version of a structure was displayed near the tablet, and participants were asked to duplicate this structure in the environment.

For the *numeric input task*, participants were asked to assign specific values to the counter, distance, and direction variables. An example task was "Generate a new structure that has three copies along the x axis, 4 copies along the y axis and 3 copies along the z axis. The distance between adjacent copies should be 10 units along the x axis, 30 units along the y axis, and 5 units along the z axis. The direction of the cloned objects should be along the directions of +x, -y, and –z."

The first task is closer to the real-world tasks. We used the second task because it still allowed us to find usability problems, while also requiring more precision. Due to the difficulty in setting up extremely precise numbers using spinners and widgets, we asked participants to complete the task within an error bound of +/-1 unit for each "spacing" variable. Tasks were read aloud for the participants and were also displayed on the bottom of the screen for their reference.

The virtual world initially contained a single-story building with four beams and four columns. The size of the beams and columns were 5x5x20 units along x, y, and z directions, respectively. The counter widgets on the tablet had a maximum value of 5, and the distance widgets had a maximum value of 60 units. Clicking on the arrow button increased or decreased the selected counter widget's value by one unit or increased or decreased the selected spacing widget's value by five units. The prototype that integrated all the interfaces was implemented using the Simple Virtual Environment (SVE) library (Kessler et al. 2000) and OpenGL.

4.5.3 Procedure

Participants were asked to fill out a pre-questionnaire about their demographic information at the beginning and a post-questionnaire at the end of the study. The experiment was conducted in a quiet laboratory setting. We tested each interface with all participants, and counterbalanced the order of the conditions to avoid learning effects.

Three participants were not trained how to use the interfaces, and were asked to explore. Think-aloud protocol was used with these subjects. The other five participants were trained and were asked to complete the tasks as quickly as possible while avoiding errors. In this way, we evaluated the system from the perspective of learning and ease of use, and also from the perspective of productivity and efficiency.

We interviewed participants and discussed the interface issues after they completed the two tasks in each interface to avoid forgetfulness and confusion about particular interfaces. This also gave participants a short break, as the experiment lasted about two hours. Behavioral data, e.g., confusion, frustration, and user comfort were noted.

4.5.4 Results and discussion

4.5.4.1 Subjective usability ratings

We asked participants in the post-questionnaire about their preferences, perceived usefulness, and ease of use on a scale of 1 to 7, where 1 was the worst and 7 was the best. The perceived usefulness was defined as "the degree to which an individual believes that using a particular system would enhance his or her job performance;" and the perceived ease of use was defined as "the degree to which an individual believes that using a particular system would be free of physical and mental effort" (Davis 1989).

We can see from Figure 4.10 that for all of these measures, participants rated the Spin, 1-DOF-O, and 1-DOF-C interfaces highly. Participants' feeling of comfort with the interface got worse with increasing DOFs, which made 3-DOF the worst interface according to our subjects.

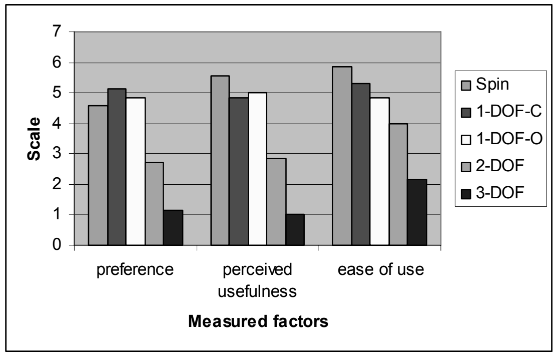


Figure 4.10. Participant ratings

4.5.4.2 Interview results

Participants were asked questions on various aspects of the interfaces, such as: How easy was it to learn? Was it easy to read the setup variables? Was the interface distracting? Was the interface cluttered? What did you feel about the feedback provided? Comments from both groups are listed below:

- Participants reported that once they learned one interface, it was very easy to use the others.

- Clicking on the arrows of the Spin interface was easier than dragging sliders on the other interfaces. But the Spin interface was not considered good for design because the spinners did not scale well; it could be impossible if an axis was required to scale into hundreds of units.

- The 1-DOF-O and 1-DOF-C interfaces were also reportedly very easy to understand and had a smaller incidence of errors. Three of the seven participants (including the two participants with construction-domain knowledge) did not like the disappearance of the third axis although they could get the invisible axis back very easily.

- The 2-DOF and 3-DOF interfaces were hard to use. Participants reported that it was very difficult to keep track of the position of the widgets for three variables. The maximum they could work on was two variables at a time. Most participants felt that using the 3-DOF interface was a distracting and stressful experience. But they also reported that they would use it if appropriate constraints were available and if less attention was needed.

- The feedback provided about the constrained moving area was enough for learning the behavior of 1-DOF sliders, but not of 2-DOF and 3-DOF widgets. Most participants did not comprehend the meaning of the rectangle on the 2-DOF interface and the box on the 3-DOF interface until they were told. Most participants did not have a problem with this feedback after being trained, but suggested that it needs to be more understandable.

- Most participants reported that the level of distraction increased with the DOFs of a widget instead of the number of widgets displayed on the tablet, and that increased DOFs also required more attention to the tablet.

- All three participants doing exploration reported that feedback offered in the 3-DOF interface was good, but they needed quite a bit of time to get used to it. However, most of the other five participants reported this interface was distracting, in that too much feedback was available on the tablet. All participants realized that they could build a structure very quickly using this interface, but fine adjustment was hard because of the higher number of DOFs available.

- Participants reported that controlling the "number of clones" widgets was easy, but controlling the spacing widget was difficult, due to the greater accuracy needed for the spacing.

- Most participants reported that they preferred to look at the parameters displayed next to the widgets compared to the scale marked on each axis on the tablet; and half of the participants did not use the scale on the axis at all.

- Participants suggested the separation of the spacing and the direction for 3-DOF interfaces due to the increased DOF. But they also said that the combination of spacing and the direction for the 1-DOF widgets was intuitive since they were fairly easy to move.

4.5.4.3 Observations

The main observation we made was that participants tended to set up one parameter at a time no matter how many parameters a widget could control. For example, when doing task 2 using the 3-DOF widgets interface, most participants dragged slider widgets along the x direction first, then along the y direction, then the z direction. The same behavior was observed while setting up the spacing variables. One participant with advanced 3D gaming experience tried to perform the task by setting two parameters at a time and then setting the third. He outperformed all other participants while using 3-DOF widgets on speed and accuracy.

4.5.4.4 Discussion

The spinner interface was easy to use because of its directness and the simple actions performed to change variables. It required less agility and manual dexterity from the user. The cognitive load of this interface was low because fewer items were displayed on the tablet during interaction. It was easy to use in that it was cognitively direct for numeric tasks: there were separate widgets corresponding to each of the nice parameters. Also, it provided sufficient accuracy for the tasks assigned to the participants. However, such alphanumeric input methods might not be efficient overall, as observed by Mine (Mine 1995) and confirmed by our experiment. All of this is demonstrated by the high perceived usefulness and ease of use, but relatively low preference.

The constrained 1-DOF movement of sliders was a natural extension of the sliders used in desktop user interfaces, and therefore put less cognitive load on the users. Also, due to their flexibility in setting up the variables, participants preferred them to other interfaces. Although dragging the slider was a continuous action that could create a fuzzy range problem while being used to specify discrete numbers (such as the number of copies, which is an integer), the participants did not complain about this.

Free-space positioning was a difficult task which made higher DOF widgets harder to control. This made the 2-DOF and 3-DOF widgets the least preferred choice among all interfaces. The combination of direction and spacing control made participants frustrated when the participant finished setting up the distance then found the directions were not right. They then had to move the widgets to another coordinate which changed the spacing again. However, we feel that we should still work to improve the usability of these higher-DOF widgets because of their advantage in speed to create objects. This must of course be handled with more interaction constraints.

So the result demonstrates that it is not only the integrated property of the input device affects the how users perform, but mobility of the user per se. Users like to perform the task in 3D since it is faster, but the user interface design does not support high precision. In a sense, it is the task requirement putting more impose on how users behave compared to the device per se.

4.5.5 Summary

Because complex but repetitive structures are common in the domains of construction and architecture, we developed five cloning interfaces that allow the generation of multiple distributed copies of objects while immersed in a virtual world. Empirical results demonstrated some usefulness of interaction techniques designed. We also find the important usability issues with regard to different design choices. The main take-away lessons from this study can be summarized as follows:

- It is possible to design some useful interaction techniques for the domain-specific task of cloning. Taxonomy represents the design space well enough to make the task easy to understand, as it matches most desktop tools.
- 2D interaction with a pen-and-tablet in an immersive VE is qualitatively different than 2D interaction with a mouse on a desktop computer. It's very difficult for users to specify two or more DOFs simultaneously with a pen and tablet interface.
- Slider widgets in a 3D interface may be better suited for discrete than for continuous numeric input.
- The attentional requirements of the interface increase with increased widget DOFs. Provide separate parameter control to reduce the flexibility of the system with high DOFs.
- Constrain the widgets if possible.

- Provide appropriate feedback, reduce cognitive load and help users easily make action plans.

4.6 Interaction Techniques for Cloning (Set 2)

4.6.1 Overview

Since we are able to create some cloning techniques, we further research on the benefits we can possibly gain by applying domain knowledge to the design of interaction techniques. We therefore follow the three-level framework to investigate its use and the pros and cons by following different paths (see Section 3.2). Also, we have used two styles to design interaction techniques either direct or indirect (pen-and-metaphor user interface). The interaction techniques, which include PORT, copy-by-example, space-metaphor, dynamic slider, keypad, are distributed in a space to cover the possible design space of cloning according to domain specificity and styles of action (Figure 4.11).

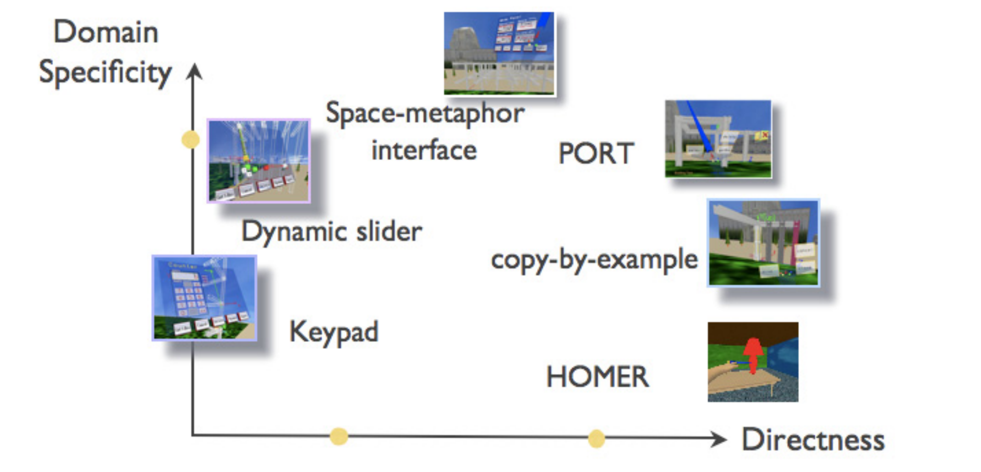


Figure 4.11. Overview of the design space of DSD for cloning techniques

For example, PORT is the most direct way to perform cloning by designing interaction techniques directly for the cloning task. Copy-by-example is a cloning technique that is tweaked from HOMER, a generic interaction technique.

4.6.2 PORT cloning technique

PORT affords not only selecting objects to be cloned, but also setting up the number of copies and distances for cloning, where users can simply point in the direction of interest and push forward/backward on the joystick. This reuse of the PORT metaphor for multiple stages of the cloning process produces consistency and therefore more intuitive operation of the interface.

Four buttons on the wand control all operations for PORT cloning. The interface displays currently available commands in a menu attached to the wand (Figure 4.12(a)). No label is presented on the upper left button to avoid extra visual search because that button is always used for travel.

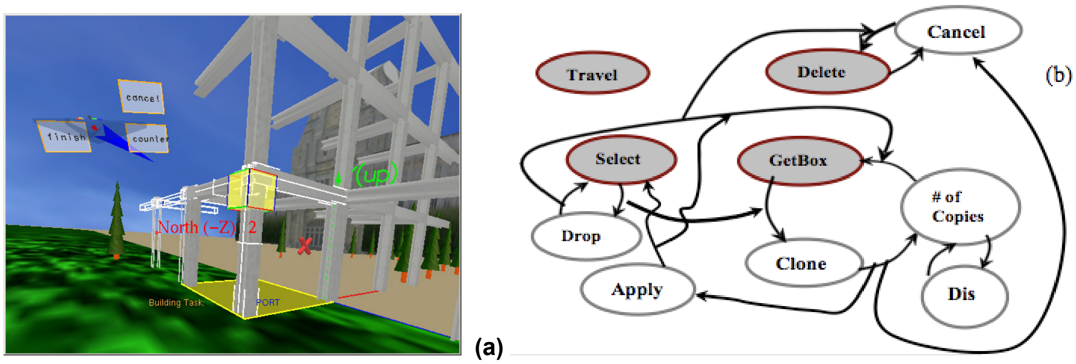


Figure 4.12. PORT cloning interface

The functions of each button changed according to the user's action. Because the cloning operation is relatively simple, we were able to accommodate all states and optimized the transition of the button states for ease of operation.

The change of buttons' states is shown in Figure 4.12(b). The initial states of the four buttons are "Travel", "Select", "Delete" and "GetBox." The "GetBox" button is a homing command for the selection box, i.e., the box is moved to a position in front of the user and its size is reset. It is useful when the user does not know where the selection box is or just wants to recover the initial state.

Users select object(s) by grabbing the box then dropping it onto object(s). This action causes the lower right button to change to "Clone". Upon selection, a bounding box appears to indicate the bounding volume of the selected objects. Clicking the "Clone" button activates cloning. The lower right button then changes to "# of copies". Simultaneously, the feedback objects attached to the selection box (the arrow and the X) move to the faces of the bounding box of the selected objects. Pointing to a direction and pushing forward/backward on the joystick will cause the increase or decrease of the number of copies. The user can click the lower right button to toggle between "# of copies" and "Dis" (to set up distances between copies) till s/he finishes or cancels the current cloning operation.

During the editing process, ghost copies of created objects are drawn to show the user what to expect. They are not added to the environment completely until the "Apply" command is issued. The parameter of the current state is displayed on the screen for a task-related cue. For example, "East(X) = 5" means that the user is pointing in the east direction while setting the number of copies to 5.

4.6.3 Copy-by-example interface

Placement of objects with a technique like HOMER-C is quite direct; however, precise placement can be time consuming if users have to travel a large distance for placing objects out of reach.

We designed a *copy-by-example interface* (Figure 3.3, Section 3.2.3) that only requires users to place the first object, and then the program automatically calculates the position of the next and creates a ghost copy there. This copy is added to the environment if another paste command is issued. Users can adjust the location of the ghost copy and the user interface automatically updates the relative location for placing the next.

This design does not consider cloning as an independent task, but rather utilizes the "repetitive" characteristic of the structure to avoid the precise object placement required by HOMER-C. It also increases the flexibility of building repetitive shapes like stairs, spiral shape object etc, which may not be possible with the other cloning interfaces. Experimentally, if we have any performance gain, we can declare that the design with domain characteristics is beneficial.

4.6.4 Space-filling metaphor interface

PORT-C only enables four actions at a time, limited by the number of buttons on a wand. It may be too small a vocabulary compared to real-world applications that demand more commands. Therefore, additional user interface elements are need for more options. We used a pen-and-tablet metaphor user interface. Four buttons, drawn on a small lightweight physical tablet (Figure 4.3, size of 25x25 cm) defined four logical modes: operation, space, preview and parameter (Figure 3.7, Section 3.2.4). Clicking the button will toggle the options as listed. The labels representing the current state of the toggle are displayed in a distinct red color, while the other labels are in black. Because the buttons are relatively large, this interface should have a quite low action transition time.

This interface was called the *space metaphor user interface* (Space-filling metaphor, Figure 3.7, Section 3.2.4) because it supports two "space" modes: fixed or varies. Under the "varies" mode, the cloning process is the same as PORT-C. Under the "fixed" space mode, however, a user-defined box becomes a constrained volume that the cloned objects can occupy. Any objects outside the boundary are automatically deleted by the system. In this case, the number of clones and the distances between clones are dependent. Changing one affects the other. We added the fixed space mode because architects think about space and scale and how to arrange objects in that space (Kurmann 1998).

4.6.5 Dynamic slider widget interface

The dynamic-slider user interface (D-slider) (Figure 4.13(a)) has seven widgets displayed on a larger tablet (size of 48x35 cm): three for the number of copies, three for distance and one for toggling the direction of copies. Four slider widgets are visible at a time and the user can toggle the direction by physically rotating the tablet. For example, moving the tablet to align it roughly with the horizontal plane will made the X- and Z-axis sliders visible. Users can pre-define the range of the slider based on the task requirement (Chen et al. 2004).

Figure 4.13(b) illustrates the behavior of the slider slot. The slider slot is divided into three parts by gap 1. The slider slot will slide to the right and scale up its scale when the projected position of the wand falls on the right side of gap 1. Moving the wand back to the left of the gap will stop the sliding so that users can start dragging the slider to set up numbers. If the wand falls into gap 1, no sliding occurs, in order to avoid the accidental movement of the wand beyond the working area of the slider.

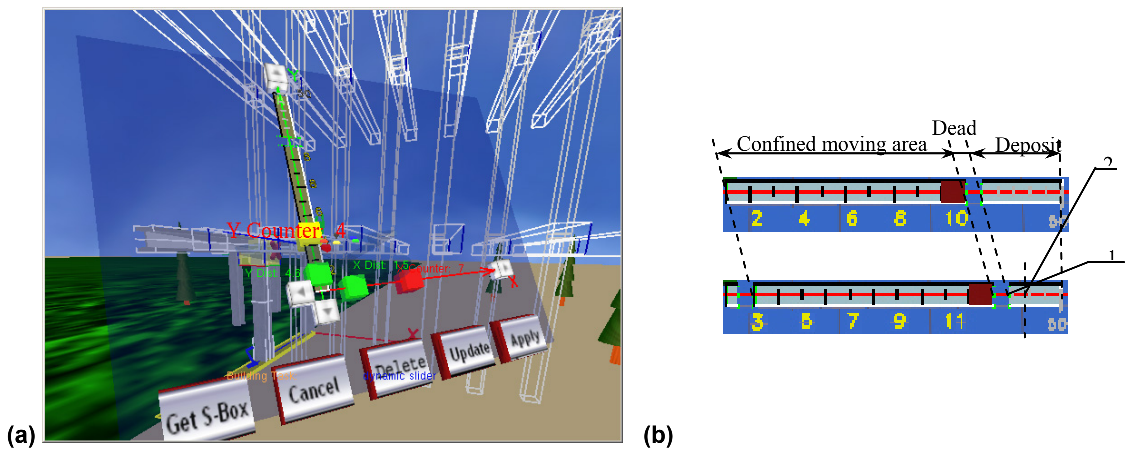


Figure 4.13. Dynamic slider user interface and slider's behavior

4.6.6 Virtual keypad

Our previous study (see Section 4.5, also in (Chen et al. 2004)) suggested that numeric input was an important complementary tool to cloning. Numeric input is also widely used in desktop tools for computer-aided design. Unfortunately, not many numeric input techniques for VEs exist. We have designed the *virtual keypad interface* (Keypad, Figure 3.4, Section 3.2.4) on the tablet to support fundamental text editing operations: insertion, deletion, clear. Users click a wand button to issue an operation. Figure 3.4 illustrates an example of setting up five copies along the Y direction. The user can also choose the parameter s/he wants to set up by selecting one of the six widgets on the tablet. This action will cause the keypad to be relocated to the position of the widget.

4.7 Summary

We have demonstrated that we are able to use domain characteristics to define a design space for cloning. We have also shown that several interaction techniques can be produced and general guidelines can be gained by studying these techniques so that practitioners can make appropriate choices.

CHAPTER 5: *Evaluating the Benefits of Domain-Specific Interaction Techniques*

This chapter describes three experiments to demonstrate the effects of domain-specific design on the *interaction techniques* for cloning and on the *usefulness of a 3D application*. This would allow us to get a complete picture of how well DSD works and how domain knowledge plays a role in the design of interaction techniques.

5.1 The Effects on Interaction Techniques

5.1.1 Experiment I: Effects of the Domain Task

5.1.1.1 Experimental Goal

The goal of this study is to examine if the addition of the cloning task would affect the work that engineers perform in their domain. We therefore compare a domain-specific task, cloning, to generic manipulation tasks since cloning was originally decomposed into manipulations. Also, most user interfaces are tested on the basis of *usability* (effectiveness, efficiency, and satisfaction) alone. In this experiment, however, we are evaluating *usefulness*, i.e., how well the interaction techniques can support real-world use.

Our hypothesis was that participants would perform better with a cloning technique designed for building large structures than with a modeling technique based on a generic selection and manipulation technique. Users would prefer to use cloning as well.

5.1.1.2 Interaction Techniques

We chose to use the PORT-cloning (PORT-C) and HOMER-cloning techniques (HOMER-C) in this study to single out the effect of the cloning task. Both techniques use direct manipulation. The HOMER-cloning interface was based on the HOMER selection and manipulation technique (Bowman and Hodges 1997), and was similar to standard copy-and-paste: users select objects of interest, and then place copies one after another with the aid of a snap-to-grid system. The PORT-cloning technique (Section 4.6) is specifically designed for the task of cloning.

5.1.1.3 Tasks

We had three criteria in mind when we chose the task for our experimental study. First, we wanted to select tasks that are practical in real engineering applications so that the results of our experimental study can be applied to designing practical 3D UIs. Second, we intended to select a task that has the testing power to accept or reject our hypothesis. Third, we want to choose a task that has not been extensively studied in the HCI literature so that the empirical findings in regards to direct manipulation and non-direct techniques can be established on a broader base.

- Task 1 (building task) required participants to add two pieces one by one to an existing structure, then to raise the building up to four stories.

- Task 2 (numeric input task) was designed for modeling a large repetitive structure. An example task was "Select that one column and two beams in the environment. Construct a two story building such that a five by five grid is created in the horizontal plane."

- Finally, task 3 (picture matching task, Figure 5.1) required the participant to build a structure to match an existing design displayed as a two-dimensional (2D) image on the tablet. This task corresponds to the situation when the designer has finished a draft design on a paper and brought it to a VE to construct the structure.

Figure 5.1. The structure for task condition 3

5.1.1.4 Participants

Eight participants (20-27 years old, males) volunteered for this study. They were five senior undergraduate students and three graduate students whose major was structural engineering. They were familiar with structural analysis and architectural design tools and used them daily.

5.1.1.5 Procedure

The procedure consisted of two sessions. In the first session, participants were given two ETS standard psychology paper tests (Ekstrom et al. 1976). The first test was the perceptual speed test, defined as "speed in comparing figures or symbols, scanning to find figures or symbols, or carrying out other very simple task involving visual perception". Participants were asked to compare a group of numbers: two in each group and mark those that are different. The second test was the figure fluency test, used to test ability to produce new figures from a given set of elements. Participants were asked to think of different ways to add details to a design. This section lasted about 15 minutes. These tests were chosen not because they were related to VEs, but because they were related to human cognition in general. We used the tests because we wanted to learn if there was any correlation between users' aptitudes/abilities and users' task performance or preferences. We then may be able to design user interfaces that accommodate their needs (Stanney et al. 1998).

During the second session participants completed a background questionnaire, were given instructions and completed two sets of trials (one with each interface). We then administered a written survey and oral follow-up questions. Participants were allowed to take a break any time. Interfaces were counter-balanced between participants. Tasks were executed in the order specified previously. Participants were told to do the tasks as fast as they could while avoiding errors.

The pre-questionnaire recorded participants' background in architecture, engineering, work habits and game and computer experiences. During the practice tasks the participants could familiarize themselves with the system and the different interaction techniques. During the experiment, they were allowed to finish task 1, but were given time constraints for task 2 (one minute) and task 3 (one and half minutes). We limited the time in order to reduce the fatigue that may be associated with the HOMER-C user interface (which may require about 50 copy-and- paste operations for some of the tasks). The experiment lasted about one and half hours.

5.1.1.6 Results and Discussion

All participants completed the experiment. Participants' performance and subjective ratings suggested that the cloning interface overall produced better performance than the non-cloning interface and was particularly superior for task 2 when a large structure needed to be built. This conclusion was drawn from the following analyses: (1) for task 1, a single-factor Analysis of Variance (ANOVA) on interaction techniques was performed. Interaction technique was not significant ($F(1,15)=2.64$, $p=0.14$). However, the overall task completion time for PORT-C (mean=68.4, stdev=20.2) was faster than HOMER-C (mean=82.3, stdev=26.5); (2) For task 2, participants' performance was measured by the number of objects created in a given time period divided by the total number of objects required upon completion (50). PORT-C (mean=93%) was about four times more effective than the HOMER=C interface (mean=23%) and this difference was significant ($F(1, 15)=76$, $p<0.0001$); (3) The same calculation was used for task 3. PORT-C (mean=95%) was 1.7 times faster than HOMER-C (mean=56%) and this difference was also significant ($F(1, 14)=11.7$, $p<0.01$).

A single factor ANOVA on the participants' rating indicated that the difference in participants' preference was significant only for the numeric input task ($F(1, 15)=57$, $p=0.0001$; mean = 6.5 for PORT-C; mean = 2.25 for HOMER-C). The difference was not significant for task 1 (mean=6 for PORT-C and mean=4 for HOMER-C) and task 3 (mean=5 for PORT-C and mean=4.5 for HOMER-C). The differences in perceived usefulness and perceived ease of use were not significant for any of the task conditions. Participants did rate the PORT-C interface higher, though two out of eight participants preferred HOMER-C to perform the building and matching tasks.

Participants commented that HOMER-C is intuitive, but that object placement is difficult. PORT-C is suitable for building large structures. Also, there is no need to place objects directly with PORT-C, which can be difficult when objects are far away and movement is sensitive to hand movement.

From the cognitive and perceptual ability tests given to each subject, the perceptual speed test scores showed high correlation with task completion time with the PORT-C interface ($r=-0.82$, $p=0.01$) but not with HOMER-C. This may have confirmed that people with higher perceptual speed had better performance (Allen 2000). The reason no correlation was found with HOMER-C might be that participants spent more time on object placement or travel which did not require perceptual speed.

Participants' self-rated game experience was also correlated with task completion time ($r=-0.69$, $p=0.05$). Their self-rated computer experience did not correlate with task performance ($r=0.58$, $p=0.13$). No correlation was found significant between task completion time and the figural test score.

5.1.2 Experiment II: Effects of the Domain Characteristics

5.1.2.1 Experimental Goal

There are two fundamentally different approaches to design domain-specific interaction techniques for cloning. The first one is to take an interaction technique at the *generic level* and couple it with domain characteristics. From experiment I, we learned that object placement was the major bottleneck; we therefore tweaked HOMER-C to be a copy-by-example technique (see Section 4.6) considering the repetitive pattern of the structures to be built. The second approach is to design from scratch at the *domain level* using a well-defined design space for cloning by specifying parameters, such as the number of copies and distance between copies (Chen et al. 2004). PORT-C is an example result of this approach. This experiment was designed to compare these two approaches, and had three hypotheses.

Hypothesis I: The use of domain characteristics will improve the usefulness of the user interface for cloning.

Hypothesis II: Users will prefer techniques designed at the domain level because it matches the users' mental model of how the work should be done.

Hypothesis III: We had already designed two types of cloning techniques: some used the pen-and-tablet metaphor while others used direct manipulation. We hypothesize that direct manipulation will outperform the pen-and-tablet metaphor user interface for cloning tasks, due to rapid transitions between actions.

The pen-and-tablet metaphor user interface takes advantage of two-handed asymmetrical input and a well-constrained 2D surface to perform tasks 3D. Users know what they can do and easily "recall" their actions since widgets are visible. A major disadvantage is that the selection time of a widget could decrease user's performance. However, direct manipulation input may not have such overhead if designed appropriately.

5.1.2.2 Interaction Techniques

Interaction techniques are chosen covering the overall design space of cloning, indicated in Figure 4.2. We evaluate them all because this will give us an entire picture of how the framework supports the generation of interaction techniques and the difference of the paths.

5.1.2.3 Experimental Design

This study had a 2x5 (Groups: G1, G2 x Interface: PORT-C, Space-metaphor, D-slider, Copy-by-example, Keypad) mixed design (Table 5.1). It would be ideal to use a complete within-subjects design for comparison purposes, but we sacrificed this to avoid the after-effects associated with prolonged use. Group 1 participants used PORT-C, Space-metaphor and Keypad, while group 2 used the Space-metaphor, D-slider and Copy-by-example. Each group was a within-subjects design, but group was a between-subjects factor.

Table 5.1. Experimental design

	PORT-C	Space-metaphor	D-slider	Copy-by-example	Keypad
G1	8	8			8
G2		8	8	8	

Sixteen student participants volunteered for this study (age 20-32 years, 15 male and one female, 12 in architecture and 6 in engineering (two were double-major). All students used structural analysis and/or architectural design tools daily. Two of them had used a wand previously for navigation in a CAVE environment. Participants were assigned to one of the two groups randomly.

The experiment procedure and tasks were the same as Experiment I, but the third task (picture matching task) that was not tested with the keypad interface. The reason was to avoid fatigue and because the keypad was designed as a complementary tool rather than a primary design tool. All participants in this study were requested to finish all three tasks and were asked to finish each as quickly and as accurately as they could. The order in which participants used the interfaces was randomized within each group to minimize any order effects. The experiment lasted about two to two and half hours.

5.1.2.4 Task performance

All participants finished the experiment except one who dropped due to motion sickness. Again, two sets of trials were performed, but the data from the first trial was not used for statistical analysis.

The results in terms of the overall task completion time and cloning time for the second trial are shown in Figure 5.2. The horizontal axis corresponds to the three tasks participants performed. The vertical axis is the task completion time for overall performance or cloning time. The time for cloning is the accumulation of all cloning operations that started at the time participants clicked a button to indicate cloning until the time another command was issued to finish the cloning operation. We first performed a between groups comparison on task completion time. Results did not show a significant difference between these two groups (F(1, 112)=3.06, p=0.1).

We performed a two-factor general linear model (GLM) procedure on interaction techniques and tasks. For overall task completion time, (1) interface was not significant (F(4, 123)=0.77, p=0.83), (2) task was significant (F(2, 123)=9.4, p=0.0042), and (3) the two-way interaction of interaction technique and task was not significant (F(4, 123)=1.17, p=0.34).

For cloning time, (1) interaction technique was significant (F(4, 123)=9.05, p<0.0001), (2) task was not significant (F(1, 123)=0.17, p=0.68), and (3) the two-way interaction was not significant (F(4, 123)=0.55, p=0.69).

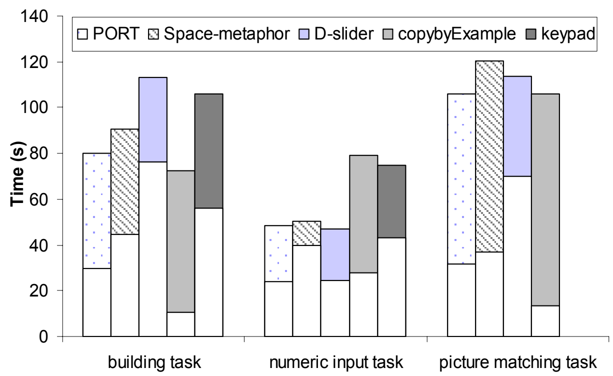


Figure 5.2. Task completion time (bottom bar is cloning time; complete bar is overall task completion time)

We then performed a post-hoc analysis using a Tukey test on task completion time and cloning time for each task. All significant factors are listed in Table 5.2 and detailed below in this section. In this table, symbols starting with A represent overall task completion time and symbols starting with C represent cloning time. Each symbol followed by a number, which is the task number (1 is the building task, 2 is the numeric input task, and 3 is the picture matching task.) The cell location indicates the two interfaces that were significantly different, with the interface labeling the column being significantly faster than the interface labeling the row. For example, C3 in the fourth column, second row means that copy-by-example was significantly faster than space-metaphor on cloning time for task 3.

For task 1 on overall completion time, interaction technique was significant (F(4, 25)=3.64, p=0.018). A post-hoc analysis using a Tukey test on differences of Least Squares Means showed that copy-by-example was significantly better than the slider interface (t=2.34, p=0.02) and PORT-C was significantly better than D-slider (t=2.03, p=0.04).

For task 1 on cloning time, interaction technique was significant (F(4, 26)=11.9, p<0.0001). A post-hoc analysis using a Tukey test showed that PORT-C was significantly faster than D-slider (t=3.85, p=0.0004) and keypad (t=2.27, p=0.03); Copy-by-example also significantly outperformed Keypad and Space-metaphor (t=3.63, p=0.0008 and t=3.04, p=0.004 accordingly). If we examine the trend of all significance, we found that techniques designed with higher directness are mostly better than lower ones (Figure 5.3). This could be explained that this task does not give advantage to cloning and the task is a mingle of a simple object manipulation task and a cloning task.

Table 5.2. Significant differences between conditions

	PORT-C	Space-metaphor	D-Slider	Copy by example	Keypad
PORT-C					
SM				C1, C3	
D-Slider	A1, C1 C3	C1, C3		A1, C3	
Copy by example	A2	A2	A2		
Keypad	C1, A2	A2	A2	C1	

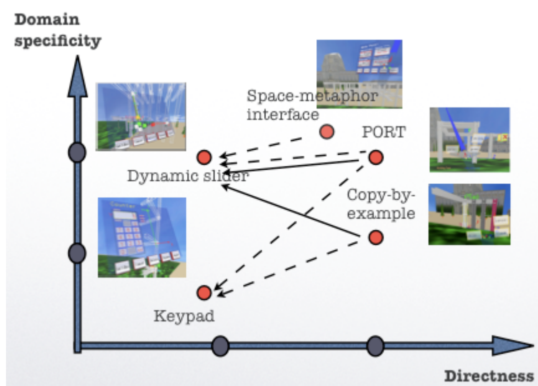


Figure 5.3. Techniques designed with higher directness are better than lower ones for the task 1.

(The arrowed lines indicate statistical significance regarding task completion time. The technique the arrow is from is significantly better than the one the arrow is pointed to. The dotted lines stand for the comparison of cloning time. The solid lines stand for the comparison of overall task completion time.)

For task 2 on overall task completion time, interaction technique was significant (F(4, 26)=6.35, p=0.001). A post-hoc analysis using a Tukey test indicated that all user interfaces with specific consideration of the cloning task outperformed the other interfaces. This was based on the following observations: (1) PORT-C was significantly better than copy by example (t=2.5, p=0.02) and keypad interface (t=2.17, p=0.03); (2) Space-metaphor was significantly better than copy-by-example (t=2.58, p=0.01) and keypad (t=2.28, p=0.02); (3) D-slider was also significantly better than Copy-by-example (t=2.5, p=0.02) and Keypad (t=2.19, p=0.03). There was no significant difference among PORT-C, Space-metaphor, or D-slider.

For task 2 on cloning time, interaction technique was not significant (F(4, 44)=1.52, p=0.2). For task 3 on overall completion time, interaction technique was not significant (F(4, 17)=1.6, p=0.2). If we examine the trend of all significance, we found that techniques designed with higher specificity are mostly better than lower ones (Figure 5.4).

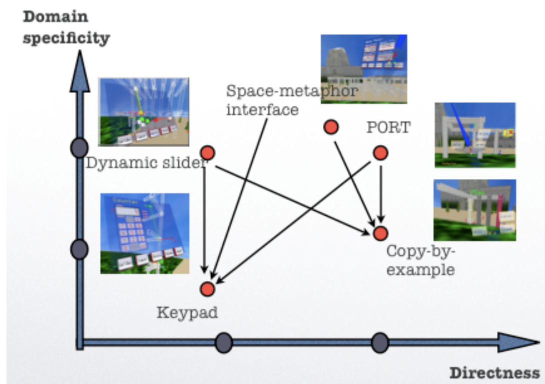


Figure 5.4. Techniques designed with higher specificity are better than lower ones for the task 2.

For task 3 on cloning time, interaction technique was significant (F(3, 17)=7.3, p=0.002). A post-hoc analysis using a Tukey test indicated that Space-metaphor, PORT-C, and Copy-by-example were significantly better than D-slider (t=3.27, p=0.003; t=3.37, p=0.002; t=4.39, p=0.0001 accordingly). Copy-by-example was significantly better than Space-metaphor and D-slider (t=2.07, p=0.046; t=4.39, p=0.0001 accordingly).

5.1.2.5 Subjective ratings, comments and behavior

There were no significant differences for preference rank and perceived usefulness among the interfaces. However, participants did rank PORT-C, Space-metaphor, and Copy-by-example highly (PORT-C: 6.1, Space-metaphor: 5.4, Copy-by-example: 5, Keypad: 3.3, and D-slider 3.2).

Participants commented that they mostly preferred Copy-by-example for simple tasks and PORT-C or Space-metaphor for complex modeling work. When participants created one element, they perceived that the task required a single object copy and placement rather than a cloning operation. Participants rated Keypad low because it required too many clicks and rated D-slider low because the difficulty of using the wand to drag sliders.

Participants preferred the smaller tablet compared to the larger one (used for D-slider) unanimously because it was light and easy to hold with a flexible grasping posture. These results may imply that we can even go further and use the hand rather than a tablet as the interactive device (Kohli and Whitton 2005).

We again found a strong correlation of self-rated game experience with task 1 completion time (r=-0.3, p=0.03), but game experience had no effects on subjects' preferences for user interfaces (r=-0.12, p=0.4). This would suggest that the game player might not care very much about the style of the user interface since they could master most interfaces quickly. Computer experience was correlated (r=-0.47, p=0.006) with task 2 completion time, but game play was not ($r=0.23$, $p=0.19$). Additionally, no correlation was found for task 3 with regard to either computer experience ($r=-0.1$, $p=0.5$) or game experience ($r=0.2$, $p=0.2$). We may infer that that computer and game experiences were associated with performance under certain task conditions only.

Fatigue was observed due to prolonged use. Most participants asked if they could sit down and work. They also avoided looking around with the HMD on and put down the tablet or changed to a different grasp. They reported fatigue associated with pointing gestures with PORT-C. These results suggest that we may need to consider the design of HMD interfaces suitable for seated use.

5.1.2.6 Discussion

Simply by considering repetitive patterns in building structures (a domain characteristic), Copy-by-example had a significant performance gain over HOMER-C (from experiment I), thus supporting our first hypothesis. It also outperformed most other interfaces under task conditions 1 and 3. The reason was mainly the ease of creating more copies due to repetition. By our observation, participants quickly placed the first copy (with snap) and then made several button clicks to create more. Participants did comment that the placement of first group was hard, but the cloning operation was exceptionally easy. With other cloning user interfaces, participants were very careful while editing numbers with joystick to avoid overshooting. It took longer to make fine adjustments. This is confirmed by the performance data with significant differences in cloning time for task 3.

Examining Copy-by-example carefully, we found that the bottleneck for its low performance on task 2 was the larger number of selections participants had to perform. All other interfaces required selecting objects once for this task, but Copy-by-example required at least three selections. We evaluated the selection time on task 2. A post-hoc Tukey test indicated that selection time for Copy-by-example was significantly longer than all other user interfaces: (PORT-C: $t=2.43$, $p=0.02$, Space-metaphor: $t=3.82$, $p=0.0005$, D-slider: $t=3.72$, $p=0.0006$, and Keypad: $t=2.09$, $p=0.04$.) Users disliked using Copy-by-example for task 2 because the others "made more sense", as noted by several participants. This supports hypothesis II regarding the cloning task.

PORT-C, in most cases, outperformed the other user interfaces, perhaps because of its simplicity and directness. Hutchins, Hollan, and Norman (Hutchins et al. 1985) coined the term "directness", and characterized it into two aspects: distance and engagement. Engagement in direct manipulation means the user is engaged with the objects themselves. So PORT-C may have higher engagement while being used for cloning and lower turn-around time because no operating on widgets is required. This result supports hypothesis II regarding the cloning task (PORT-C is a technique designed from scratch at the domain level), and also hypothesis III (PORT-C, which uses direct manipulation, generally outperformed other techniques designed at the domain level, which use the pen-and-tablet).

Widgets were implemented as part of the pen-and-tablet user interface for D-slider, Keypad and Space-metaphor. Using the pen-and-tablet metaphor added at least two extra time costs: a "transitional time" between the tablet and the 3D world and "operational time” on widgets. The reason that the D-slider was slow might be the high cost of “operational time”. This is at least partially due to the input device (a wand) we used and its lack of support for the pen-and-tablet style. A stylus or pen-like device would clearly be better. A few participants held the wand like a pen during the experiment when operating on the tablet. Also, jitter made the selection of sliders difficult though target expansion was adopted.

The disadvantage of the pen-and-tablet metaphor, however, seemed diminished in Space-metaphor. Cloning time was significantly shorter than D-slider for task 1. And no significant difference was found between PORT-C and Space-metaphor. This suggested that hypothesis III was true. It was because the operational time was minimal so that the only tablet-related cost associated with the space metaphor interface was the transition time. The space metaphor interface used large buttons that were easy to select and click and avoided precise manipulation and jitter of the wand. In fact, one participant mentioned that he felt like he was using a touchscreen on the tablet.

The “touchscreen” interface also differs from D-slider and Keypad in how users switched their attention. We observed that participants “causally” clicked on the large buttons and switched back to the world to continue their work. This had a negative effect, however, because participants were unlikely to check the current state of the interface, producing high error rates.

5.2 The Effects on the Virtual-SAP application

5.2.1 Experiment Goal

The purpose of the DSD approach is a pragmatic one – to guide system design in order to increase usability and usefulness. To verify that DSD does indeed produce these benefits, we have performed an empirical evaluation, studying real-world tasks important to real-world practice in our chosen domain.

The aim of the study was to provide extra evidence that a 3D UI designed with domain-specific interaction techniques (produced from DSD) increases the usefulness of a real-world application (Virtual-SAP) in the AEC domain, and that overall users’ experience is improved (Alben 1996). We did this by comparing two implementations of Virtual-SAP: one with domain-specific interaction techniques for the task of cloning and the other without.

Our previous experiments have evaluated the cloning techniques and compared techniques produced by following different design paths. Participants in those studies were required to use the cloning techniques, and usability was measured with standard metrics (e.g. time, accuracy). They are not given alternatives to finish a task. In this study, participants in the domain-specific group had the choice of using the cloning techniques or other techniques for building a structure. If we found that users make use of cloning techniques to improve their work efficiency while other techniques are available simultaneously, we can then declare that techniques produced by DSD increase the usefulness of this application, and that designing at the domain level has merit.

5.2.2 Virtual-SAP User Interface

Widgets displayed on the tablet include several "tabs," similar to many desktop interfaces, which contain sets of related controls (Figure 5.5). For example, the simulation tab (Figure 5.6) displays the progress of the simulation. The New tab contains controls for defining new buildings, basic structural elements (Figure 5.8), and new structures (Figure 5.7). The Edit tab contains all the modeling actions the user can possibly do and all parameters associated with the selected action (Figure 5.5).

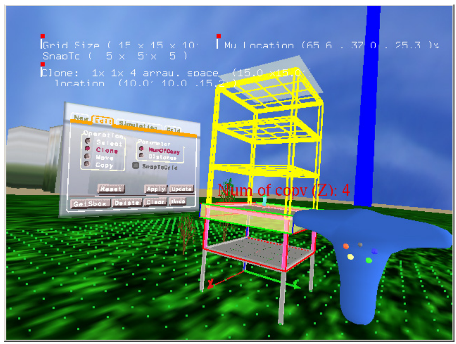


Figure 5.5. Virtual-SAP user interface including a tablet, heads-up displays, and a virtual Wand

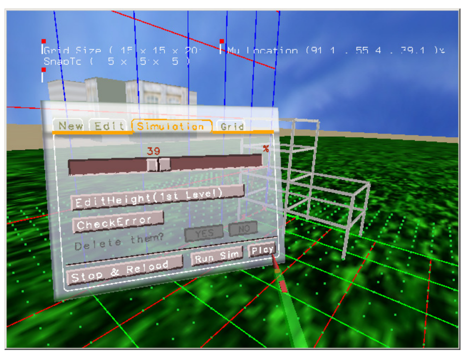


Figure 5.6. Simulation tab with the slider bar displaying the progress of the simulation

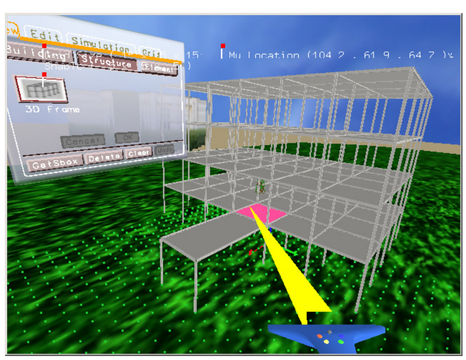


Figure 5.7. New tab showing new structure

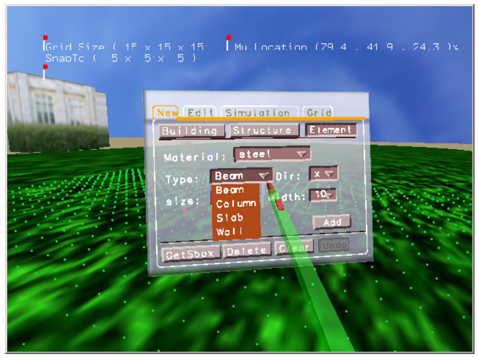


Figure 5.8. New tab showing new elements

The difference between the two user interfaces compared in our experiment was in the presence or absence of cloning operations. In the domain-specific group, cloning operations existed in the "New Structure" tab where a button labeled "3D Frame" allowed the user to create an open-frame structure or Beam-slab structure from scratch (Figure 5.7). Cloning was also supported as an operation that could be combined with other operations of selection, movement etc (Figure 5.8). These two functions were not available to the generic user interface group.

In all other respects, the two user interfaces were exactly the same; thus any differences the experiment found would be due to the effects of the domain-specific interaction techniques produced by DSD.

Just like we design interaction techniques following some rules, the interface was also designed following some philosophies. Since the goal lies in learning the effects of domain knowledge, we skip the detailed design rationales behind interface design. We only show two examples here so readers will know that every interaction technique has been well tuned to its best.

First, because 3D UI is relatively new to the users, users' strategies in doing things varies from person to person, it is useful to design an UI that introduces good strategies so they can solve problems more efficiently. This can be done through training and through changing the UI's behavior. For example, the most effective means to use the pen-and-tablet metaphor is to slide the pen on the tablet similar to use a pen to draw on a piece of paper. However, users generally would operate at a distance because they won't be able to see the physical world and they worried if putting two hands too close would damage the device. To get their dominant hand closer to the tablet, we change the shape of the device from a wand to a pen to indicate its use. In fact, users cannot click on the buttons if the visual feedback is not a pen.

Second, another design feature deserved to mention is that an open-frame button is used that is similar to desktop SAP that uses a representation both for designing an open-frame structure (as of problem) and for the cloning (as of the solution). This is because while people use visual features and physical constraints in solving problems, such as in comparing objects and remembering situations, people tend to search what they know to solve a problem. In this way, users can reflect on their mental images, lessening the cognitive load of remembering mental images if the UI design represents the problem and the solution. There are many other design considerations to make users easier to perform in 3D, such as showing where the cursor (a small red ball) is on the tablet, etc.

5.2.3 Participants

We recruited 12 graduate students with various backgrounds in the AEC domain, including 8 males and 4 females. Three participants had a building construction background; one was from architecture; and the remaining eight participants were from the civil engineering department and associated with the structural engineering group. None had significant prior experience using immersive VEs, but all were experienced computer users (on a daily basis) and all were experienced users of at least one computer-aided design tool, such as AutoCAD®, SAP®, SketchUp®, or a general modeling tool such as 3D Studio Max. Participants were assigned to either the domain-specific group or the generic group. Participation was voluntary. Each participant was paid $10 per hour for his or her time.

5.2.4 Protocol and Measurement

To evaluate the two user interfaces in terms of usefulness, a new protocol and set of measures was used that combined elements of qualitative and quantitative data collection. Both naturalistic and performance measurement methods were used. Observational data and subjective comments suggest the effectiveness of the systems; and performance measurement (e.g. task completion time, error rates) indicates efficiency. Specifically, data we collected included the following:

- Verbal communication: We used the Bales and Forsyth classifications (Bales 1976, Forsyth 1999) of user behaviors in our experiment (Table 5.3). User behaviors are classified into two types: task-oriented and social-emotional behaviors. Each type has several sub-types as shown in Table 5.3.

Table 5.3. Classification of user behaviors

Behavior		
Task-oriented		Asks for opinion
		Ask for suggestions
		Ask for information
		Give suggestions
		Give opinions
		Give information
Social-emotional Behavior		
	Positive	Shows solidarity
		Dramatization
		Agrees
	Negative	Disagrees
		Shows tension
		Shows antagonisms

- Drawings: We asked users to draw building structures that could possibly be designed using their assigned user interface.

- Task performance: This was used to measure efficiency. We hypothesize that users using the domain-specific user interface would outperform those using the generic UI.

- Comments: We asked users questions to clarify their thought processes and rationale.

The study had three sessions, with the first session lasting 1.5 hours and the other sessions lasting 1 hour each. The three sessions were held on three consecutive days, which simulates the proposed real-world use. Also, this allowed us to measure "expert" performance in the final session.

5.2.4.1 Session I

During the first session, we organized the participants into six pairs for group training purposes. We used group training because we had observed in several instances in pilot studies that participants learned quickly by talking to or observing their peers. Learning is built up through conversation between persons because they construct meaning and state their differences. Therefore common ground and mutual knowledge are established through successive turns of action and talk (Clark and Brennan 1991). We also observed that through conversations, experienced persons in a field pass along their knowledge to others by answering questions and discussing information (Garfinkel and Sacks 1970). During this session, we gathered verbal communication data.

In the half-hour training section, participants were told about the purpose of Virtual-SAP and the operation of all of its functions. For the remainder of the session, they were asked to construct some buildings of their own choosing and to help each other. During this section, participants were allowed to ask the experimenter or his / her peer questions. Participants took turns with the HMD, switching every 15 minutes so that everyone had opportunities to practice.

5.2.4.2 Session II

At the end of this session, participants were randomly assigned to either the generic or domain-specific group. They were then given a homework assignment to draw three building structures by hand. The structures were to be realistic, of interest to the participants, and buildable using the assigned Virtual-SAP interface. They were told to bring their drawings for the second session.

During the second session, participants used Virtual-SAP to create 3D models of their own drawings. Again, they were allowed to ask the experimenter questions. The goal of this session was to allow participants to practice until they reached an expert level. Also, this session allowed us to gather comments about the use of the assigned interface for these real-world tasks. So when the users use the interface to perform design tasks, users behavior can be considered as an externalization process (Schon 1983). Different externalizations result in variations in problem-solving strategies and performance because such representations change the nature of the problem. A think-aloud protocol was therefore enforced.

We use drawings because it is a means not merely externalizing a designer's mental images, but also an interpreting and understanding a situation. A designer produces a representation, which is not always associated with a pre-assigned meaning. The meaning is generated through interpretation as the designer interacts with the representation. Combined with section one when participants reflect during the act of externalization by think-aloud, this section resulting externalized representation of the UIs users used, similar to the concept of "reflection-in-action" (Schon 1983). Thus, drawing is a means of interpreting and understanding the design of 3D interaction.

5.2.4.3 Session III

During the third session, we collected performance data based on benchmark tasks (Figure 5.9). Participants were asked to build three structures and were asked to construct the buildings as quickly as possible. The buildings were chosen from participants' drawings from the generic group. They were relatively complex and could be built in a reasonable time using both interfaces. We chose tasks that were practical in real architectural applications so that the results of our experimental study could be applied to the design of practical interaction techniques. We did not ask participants to build structures with very large numbers of structural elements because that would give obvious advantage to cloning and because such tasks had already been studied in our previous experiments.

The target structure for each task was displayed in the virtual scene; participants had the freedom to construct the structure anywhere in the virtual space. The first building had a repetitive pattern. The second building had two bays of different sizes. Participants had to construct individual slabs of different sizes (by changing the size of the grid) and then apply cloning or copy-and-paste to build the rest. The final structure was a small building with a high-ceilinged ballroom on the top floor. A balcony on one side does not cover the whole span of the building so it looks like one corner is missing from the building.

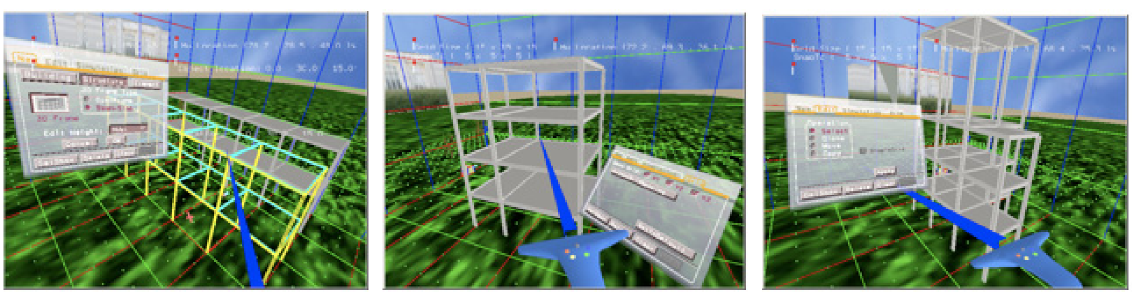


Figure 5.9. Task conditions

5.2.5 Results And Discussion

Ten participants finished all three sessions of the experiment. Two participants were dropped: one (assigned to the generic group) was too busy to continue this experiment; the other (assigned to the domain-specific group) was sick (temperature, not motion sickness) and could not continue. This left five participants in each group.

5.2.5.1 Verbal Communication

A summary of the verbal communication results is shown in Table 4. Much of the verbal communication was task-related, such as "can you change the horizontal span to 7 along the X axis direction?" While using the generic user interface, participants spent more time discussing how best to place the objects and the order of placing them. One pair actively collaborated in order to build a structure. Below is their conversation for the use of both interfaces when they were making changes to an existing structure.

Table 5.4. Participant behaivors

Behavior			Generic-group	Domain-specific group
Task-oriented		Asks for / give opinion	Focused on discussing the order of placing structural elements (beams / columns / slabs) in a structure	Quickly discuss the order of operation; and start doing things
		Ask for / give suggestions	(Very few because they were focusing on the acting in VEs)	Shall we add a horizontal span?
		Ask for / give information	"How should I change the size of beam?"	(Very few, because it is easy to do things.)
Social-emotional Behavior				
	Positive	Shows solidarity	Yes	Yes
		Agrees		Two participants said they were glad that they were in the domain-specific group so that the testing feels more "real" to them.
	Negative	Disagrees	One participant requested to stay after his section was over to observe a domain-specific group participant performing the same task. He said he wished he were assigned to the domain-oriented group. That would make him more efficient in doing the task.	
		Shows tension		

While using the domain-oriented user interface:

P2: "Can you change the horizontal span to 7 along the x direction?
P1: "Yes.." (started performing it and got it done in half a minute)
P2: "um.. this looks pretty good, but we should add a section in front of the building similar to the structure in our last our homework. Can you select the part in the middle and clone them to add a bay? This will do the work."
P1: "Yeah.. how about adding two more bays? We can check something new."
P2: "Sure….it is easy to remove things if we want to … "

While using the generic user interface:

P1: we can add two columns first, then two beams, then a slab. (long thinking time trying to figure out what to do next…)
P1: "Ok.. I would have to move the column since it is not at the right location. I will move this column from here to there." [he was acting on the object.]
…… [long wait for the participant to get his work done]
P2: "hey, maybe it is better to move that column first…"

Both groups exhibited both task and social-emotional behavior, but the participants in the domain-oriented group presented more task-oriented behavior than the generic group.

Participants in the domain-oriented group were more likely to focus on the analyses of the building structure and to link the modeling process to their own work. Their conversation was mostly task-oriented. The generic group had more complaints about the slow travel speed because sometimes they had to carry objects in order to place them at the desired position. The domain-oriented group had less travel time therefore they did not notice the travel problem until they started building large structures and trying to move out to gain an exocentric view.

After being assigned to the generic group, one participant said he wished he were in the other group so that he could build real structures for his thesis work and watch the vibration while immersed. This participant behavior indicated that the user interface, though simple, is still useful for real work. For both interfaces, participants gave high scores when asked about the operational usability issues.

5.2.5.2 Drawings

Participants in the domain-specific group were likely to draw a wide range of structures, including both simple and complex building structures (Figure 5.10). One domain-specific group participant brought one of the structural designs for his thesis project and constructed the structure using our system. This design included some curved shapes that he would not be able to create using our system, but he said he believed that the future system could handle it given the fact this is the most advanced 3D environment he had seen (he was a participant of some other experiments on campus as well).

Participants in the domain-specific group especially tended to draw larger structures regarding the size of the building and the numbers of elements. Participants from the generic group tended to draw simple structures instead. The fourth drawing in the generic group contains various room sizes. We asked that participant why he drew that; he said it was an office space, and that since he did not have access to cloning, he tried to make the experiment interesting. This was an indication that even with the generic interface, participants were engaged when using the system. It concurred with our previous study that even the simple user interface had good usability and a certain level of usefulness. This particular participant also commented that he would love to clone the first level as opposed to using the copy-and-paste function multiple times.

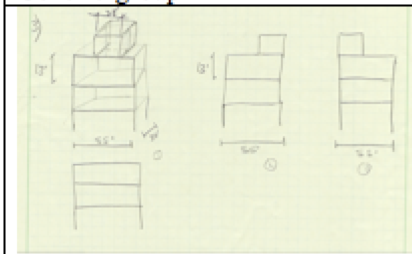
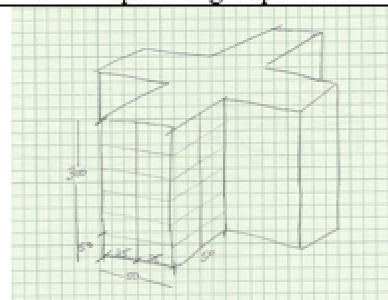
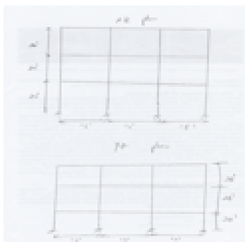
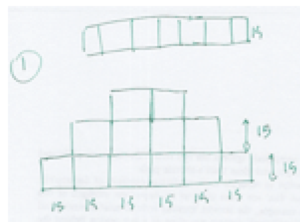
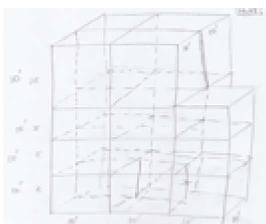
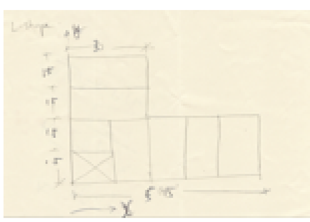
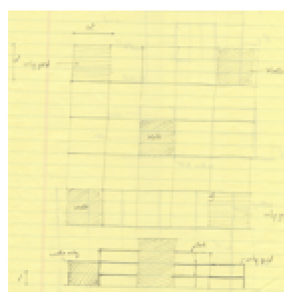
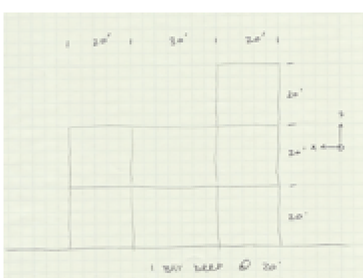
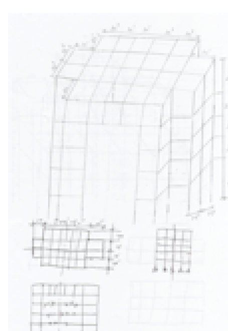
Generic group	Domain-specific group
	
	
	
	
	

Figure 5.10. Selected participants' drawings

5.2.6 Task Performance

Figure 5.11 shows the task completion time of all participants for the three tasks in session III. The domain-specific group had better performance than the generic group. The mean of the overall task completion time in the domain-specific group is about 4.3, 2.3, and 2.6 times faster than the generic group for tasks 1, 2 and 3, respectively. This supports our hypothesis that the domain-specific group could do the work more efficiently.

The first participant in the domain-specific group had the worst performance. This is mainly because he did not utilize cloning at all. He had trouble thinking of constructing structures with cloning due to his architectural background (he was used to building structures piece by piece). Participant 1 in the generic group performed well on task 2 since the structure for that task was chosen from his initial drawings.

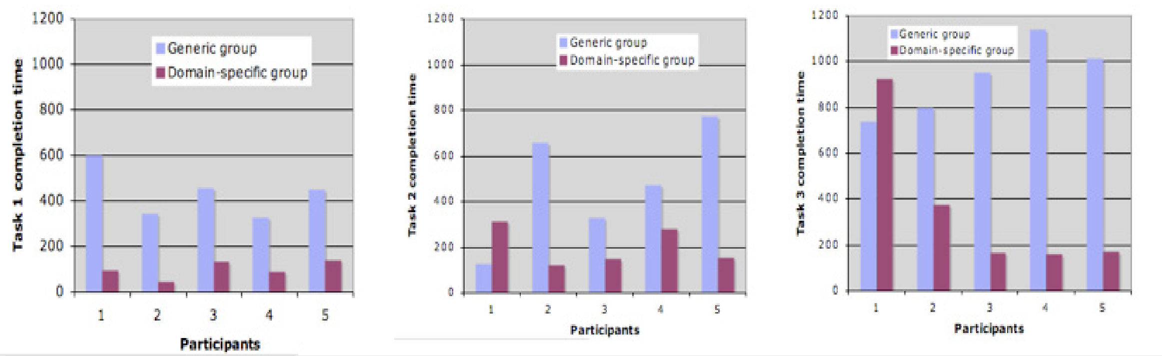


Figure 5.11. Task completion time

5.2.6.1 Participants Comments

Participants in the generic group commented that the user interface allowed them to build reasonably complex structures. The user interface was easy to use and easy to learn. All operations were greatly simplified with the snapping feature and the unconstrained action sequences we provided. However, they did comment that real building structures could be much larger, and that cloning techniques would be necessary in order to simplify the creation of large buildings. Participants in the domain-specific group commented that it was easy to create structures from scratch. The interface offered flexibility without constraining the order of operations.

No significant difference was found between the two groups with regard to general usability issues, such as ease of use, ease of learning etc. This result indicates that traditional usability evaluation metrics may not be adequate to determine the usefulness of an interface. Qualitative metrics such as observations of behavior and elicitation of user rationale are often more valuable.

When asked about the cloning operation, participants commented that it was widely used in almost all design tools and is generalizable. This comment suggests that we should be able to reuse the cloning task and techniques in other applications in the AEC domain.

Several participants suggested that VEs would be very useful for other applications in the domain, such as arranging above-ceiling components, interior design and design reviews.

5.2.7 Discussion

The results from the experiments favor the user interface produced by domain-specific design. Because the availability of domain-specific interaction techniques was the only difference between the two conditions, we can declare that the resulting differences were caused by the addition of cloning.

A likely objection to this study is that we are comparing a bad orange and a good apple, in that we added functionality to an existing design. Our answer is that the impressive benefits provided by the simple addition of one piece of functionality (cloning) to an already highly-evolved and usable VE application show that cloning is a critical task in this domain. We would likely not have considered the task of cloning, however, had we not used the DSD approach. Finding out what to design and what should be included is a crucial step that is often done inadequately (or not at all) in existing approaches to 3D UI design. Still, this study was only a first step in validating the benefits of domain-specific design.

5.3 Summary

We demonstrated that by using the DSD approach and the three-level framework, we were able to design several interaction techniques following different paths. We also performed empirical evaluations to explore the tradeoffs between different paths. The evaluations showed that, at least for this moderately large sample of techniques, the techniques with higher levels of domain-specificity were more effective. We also showed the potential benefits of the DSD approach to real-world applications, by demonstrating that a version of Virtual-SAP with cloning techniques was superior in several ways to a version without those techniques, despite the highly-evolved and usable interface in both versions of the application.

We do not claim from these studies that domain-specific techniques are always more effective than generic techniques, nor that DSD is better than other design approaches. What we have shown are some compelling and real-world examples, backed up by empirical evidence, of the potential benefits of domain-specific 3D interaction in general, and DSD in particular.

There are several major takeaway lessons from this work, three for design theory, and six in general.

- Generic interaction techniques can handle simple tasks well but are often not suitable for real-world complex task conditions; domain-specific tasks are therefore needed.

- Domain-specific interaction techniques can improve task performance profoundly; participants preferred them because they matched the preferred workflow for creating large complex structures.

- An application integrated with domain-specific interaction techniques may be more useful than an application without.

- Techniques designed from scratch at the domain level can be superior to generic techniques modified to consider domain characteristics regarding the task of cloning.

- 2D WIMP-style user interfaces can be used effectively in immersive VEs, but designers should be careful in applying them until other technological problems (such as tracker jitter and device form factor) are solved.

- Use the direct PORT cloning if the user can move in 3D dimensions; use high dimensional slider widget UI cautiously. Low dimensional UI works well if tracker jitter is not a big problem. When using them to build UIs, choose two complementary ones to accommodate users' difference.

- Display a miniature view on the focus of user's view if possible to avoid the divided attention.

- Using multiple training strategies if possible.

- Facilitate the use of magic VE interactions.

CHAPTER 6: *Reflection on the System Design Process*

There are two people in the loop for building 3D UIs: designers and practitioners. Designers are those who can provide the interaction techniques; and practitioners are those who can make use of interaction techniques produced to build useful UIs. As we have noted in Chapter 1, by the year of 1999, there are many interaction techniques produced (by designers) but the real-world use of them (by practitioners) remain small if not none. On the one hand, designers were stuck to a small set of tasks without producing more useful techniques for real-world use. On the other hand, practitioners tend to stick to a small set of well-understood techniques and are reluctant to use more powerful ones to attempt novel applications that could have benefited by using VEs.

In the first five chapters, we attempt to solve designers' problem in finding ways to use domain knowledge to handle the design of interaction techniques because knowledge is a best way for reuse. In the following two chapters, we attempt to solve practitioner's problem by charting how to design a 3D UI by choosing effective interaction techniques based on the three-level framework and the DSD approach.

Additionally, the idea of having DSD is to produce interaction techniques that are suitable for real-world use; and the user interfaces enhanced with domain-specific interaction techniques are highly useful so that users can perform their work-related tasks effectively. Motivated by our success in constructing Virtual-SAP application, we reflect on the design process we have taken to that result. Our approach is an integrated approach that combines the DSD approach and the UE process (Hix et al. 1999).

6.1 Introduction

The design of three-dimensional interaction system is a complex *problem solving process* in that a design goal is set and designers or practitioners look for solutions to solve the problems. The stopping rule could be that the user is willing to use the 3D UIs to perform some tasks and the UIs are more efficient or effective than other means.

The process can be understood as four steps: analyses (needs / tasks), design (solution search), development (implementation), and evaluation (verification of solution, criticizing bad designs, and finding new tasks and solutions). Since the system design processes often focus on what tasks users do and what systems are able to provide, we view the interface design problem as oriented towards improving the design of interaction techniques (as solution space). The end goal is to allow users to accomplish their work-related goals (as problem space) and 3D UIs are therefore more useful.

Among these four steps, the analyses and evaluation steps are relatively well studied compared to the design step. For example, scenario-base design (Rosson and Carroll 2002), ethnomethodological field studies (Button 2000), and task analysis (Johnson 1992, Vicente 1999) are widely used approaches to capture users' needs. Results from the task analyses include not only identification and description of tasks, but also ordering, relationships, and interdependencies among user tasks. These requirements are further translated into functional and physical requirements of system tasks that a user interface should satisfy.

Often the process is viewed as problem decomposition (Simon 1981, Brooks 1988). These functionalities then offer a means to form sub-problem spaces; so designers can solve the overall problem by finding solutions of these simplified sub-problems. The analyses can be done well because when we (designers or practitioners) face a design problem, the need (end goal) is often known (at least to the users). For example, the goal of medical training VEs could be to learn the anatomy or to conduct pre-surgery training (Mosegaard et al. 2006). A psychiatric treatment is targeted to recover a patient to a normal state (Pair et al. 2006). The decompositions are also somewhat known because many domains applying VEs are well trodden for architectural modeling applications; little search for decompositions is needed to conduct as part of routine design activity. For example, we can learn very much from what tasks are required by examining the functions provided by the desktop systems.

The evaluation step in the overall design process is considered critical to finding design problems. The usability engineering process (Hix et al. 1999) (Figure 6.1), while applied to VEs, is an evaluation-centered approach. This approach manifested in four stages: (1) user task analysis, (2) expert guideline-based evaluation, (3) formative user-centered evaluation, and 4) summative evaluation. Alternative usability studies are conducted to discover usability problems associated with design at different stages of development (Bowman et al. 2002). Heuristics evaluation is good to be applied at an early design stage with a few experts; formative evaluation can be used for comparisons of different techniques and interfaces. Other factors besides the evaluation methods are addressed such as evaluator's factor, device factor, and so on. Therefore, if a designer focuses on usability throughout the design process, the VE system produced can at least be usable in that there are no nonsensical user interface (UI) components (Bowman et al. 2002).

The third step, development, belongs to more software engineering side of the problem that utilizes *architectural* model (e.g., MVC (Krasner and Pope 1988), Interactors model (Myers 1990), or X/Motif toolkits (Heller et al. 1994)) for UI development. It is not the focus of this work. Also, our design process does not assume the role of or impose requirements on a program development environment. Our aim is on interface *design* rather than *development*. The bridge between architectural model and our method lies in associated tools for offering a higher-level software approach to the design of interactive systems (Szekely et al. 1993, Nichols et al. 2004).

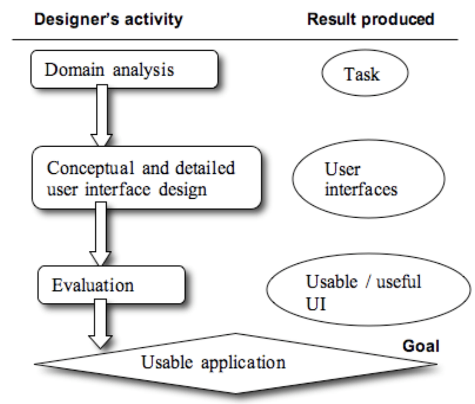


Figure 6.1. Usability engineering process

What has not been studied extensively is the design step, i.e., to provide satisfying if not optimized solutions to build usable and useful interaction techniques for the applications. UE tried to address the problem through iterative design and evaluation. The designers often *synthesize existing* interaction techniques and are reluctant to produce novel design (Beaudouin-Lafon 2000, Wolf et al. 2006). There is nothing wrong for doing syntheses; however, until we have a collection of useful interaction techniques that meet application needs, the design of interaction techniques is critically important *within* the UE process. As we have stated, there are not many interaction techniques that can be picked to address real-world needs. Also partially because of the lack of integration approach in the UE process, three-dimensional VE applications have very naïve interaction even for high-interactive architecture modeling applications.

Set against the view of design as a deliberative problem-solving relying on instantaneous and intuitive outcome that occurs from sudden burst of insight, we believe the design process of interaction techniques can be a sequence of steps designers can follow. Even when a plausible solution can occur because of the burst of insight, still reasoning of certain solution is required, for example, using design rationale (Carroll 2003), claim analyses (Sutcliffe and Carroll 1999), or case studies approaches.

Motivated by the certain level success we have gained by using DSD to design interaction and further the successful integration of domain-specific interaction techniques in the design process, we reflect on the process and propose the integrated design approach of DSD and UE process for a rigorous interface design process.

Our approach includes two threads that combine taxonomy-based testbed design and evaluation application and conventional UE process in the system design process. We separate the study of interaction techniques in the design process also because the belief (Nielsen 1993) that VE should replicate the interaction from the physical world is incomplete. Though we gain a good understanding of task space, the generation of interaction techniques could have enhanced human's perceptual and cognitive abilities to perform what is impossible in the physical world to enhance our job performance. Pure intuition from usability studies is not enough to construct 3D interfaces (e.g., the keypad for cloning is not the most effective means though it is the easiest to use from our previous usability studies).

For the first step, a designer can start with the integration of a set of existing interaction techniques (possibly large and generally designed for lower level tasks). For example, for the design of an architectural design review applications, the designers can apply Go-Go (Poupyrev et al. 1996) for navigation, HOMER (Bowman and Hodges 1997) for manipulation, and the it-viewing technique (Tan et al. 2001) for target-based travel. Primary solutions are to connect these techniques in a reasonable manner. This stage is often used in the UE process.

Our emphasis lies in the second step where the DSD is applied to improve the interaction. The difference from UE process is that it is not a pure turning of existing interface but to innovatively design new techniques seemingly more suitable for tasks at hand. According to Simon (1981), design is ill-structured. So the initial vagueness prevents the designers finding a precise problem solving space and setting clear criteria for determination of a final one. For highly interactive applications in architecture, medical data visualization, and engineering application, where high interactivity is desired, there are often some specific constraints from the application desirable to be addressed. It could be beneficial to provide a large solution space and then radically shrink it to an optimal solution through the constraints imposed by the domain.

For the second step, therefore, the design of domain-level design of interaction techniques is carried out given the usage contexts. In this step, a set of interaction techniques is designed using taxonomy-based testbed design and evaluation approach to address domain characteristics derived from domain needs. As we have seen in Chapter 5, empirical evaluation suggests that 3D UIs combining DSD and UE are more useful compared to the one designed with only one-step process, at least for the case of Virtual-SAP. Of course, the two-step design is recursive: the design of missing techniques can be undertaken in the next iteration.

Our design approach is similar to Bowman's formal design framework (Bowman and Hodges 1999) in a way of using testbed to conduct the design of interaction techniques. However, the difference is that the content presentation is isolated from the application context in Bowman whilst ours are strongly tied to the application contexts to address the real world use of the interaction techniques. Our design process is also similar to Hix's latest work of using UE process (Hix and Gabbard 2002) in a way of designing interaction techniques for the real-world use. However, the difference is that we stress the stepping back to examine the domain knowledge use and reuse as the way that interaction techniques produced can be reusable for certain tasks and for different tasks conditions. The three-level design framework is used to compare interaction techniques. In this way, the efficiency of the overall UI can be improved.

6.2 *Overview of the Integrated Approach*

Figure 6.2 illustrates the integrated domain-specific design and development process that incorporates the DSD design in the UE process. Taxonomy-based testbed design and evaluation approach is applied in our process to produce domain-specific interaction techniques.

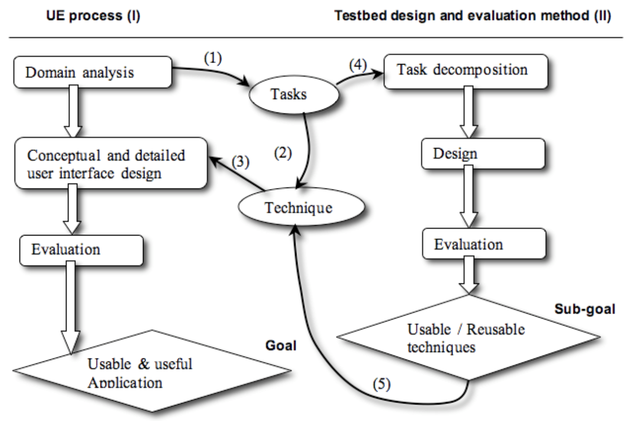


Figure 6.2. An integrated process that inserts domain-specific interaction technique design and evaluation into the UE process

Because of the independent nature of UE process and the testbed evaluation approach, this integrated process maintains the integrity of the two approaches, but coordinates design and development activities to utilize the benefits of each. It is not the goal to merge either design approach into the other, but to loosely establish a development infrastructure in which both can coexist and contribute their respective strengths to the overall design process.

The design is also medicated by tasks and techniques that are shared between these two processes.

6.3 *Step 1: Integration of existing interaction techniques*

6.3.1 Description

This step represents the conventional UE design process following path 1->2->3 illustrated in Figure 6.2. Because UE process is a top-down engineering approach, the design of interaction techniques is often conducted by a means of *re-formulation* of tasks to a sequence of sub-tasks so that existing repertoire of interaction techniques can be used. For example, the cloning operation is decomposed into multiple steps of single object selection and manipulation. The interaction techniques for objects' selection and manipulation are picked and integrated. Using this approach, designers can quickly come up with a prototype that can be used as a tool for discovering further needs.

6.3.2 Example

The first implementation of Virtual-SAP (Bowman et al. 2003) has followed a usability engineering process where Go-Go and ray-casting were applied. Task analyses were done with domain experts who are actively involved in needs collection and usability studies (Setareh et al. 2001, Setareh et al. 2005). Results from the task analysis were used for functional decomposition and composition. First, a top-down approach was used to break the high-level activity into small executable steps based on workflow analyses. Application designers then tackled sub-problems in each step individually using existing techniques. Finally, a bottom-up approach was used to adapt sub-solutions to meet the application goal. Iterative evaluations with domain experts and usability experts were carried out using different approaches.

Figure 6.3 represents task decomposition and proposed interaction techniques in Virtual-SAP. Major high-level activities are those used to create and edit models and to run simulations. The modeling task was decomposed to several sub-tasks, such as travel in the environment ("move around"), editing a structure, and loading new or predefined structures. The editing task is further decomposed into several levels before reaching selection, manipulation, and system control (for changing the sizes and types so that the right model can be loaded). The task hierarchies and the identification of different components of a real-world system provide helpful guideposts for articulating and examining the context of use and how to reach the sub-goals.

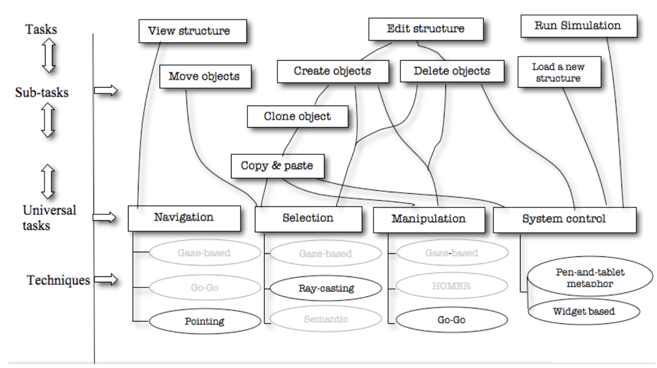


Figure 6.3. Task decomposition in Virtual-SAP 1.0

Upon reaching the universal task level, designers can then pick the best available interaction techniques and compose them into the final user interface. The interaction techniques were chosen according to the action sequences and system level consideration of the relationship between sub-goals. In Figure 6.3, the techniques shown in the ovals are some possible choices of interaction techniques. Those drawn in black were used in the original implementation while those drawn in gray were not.

The results from several user studies suggested that the interface (Figure 6.4) was relatively usable for the tasks of building small structures (Bowman et al. 2003, Setareh et al. 2005). When we gave the application to students and engineers for doing real-world structural design, we found that it was not very useful for constructing large structures with several tens of objects not to mention hundreds. Using the original implementation of Virtual-SAP, users would need to spend hours to move hundreds of beams and columns in order to design even a moderately complex structure.

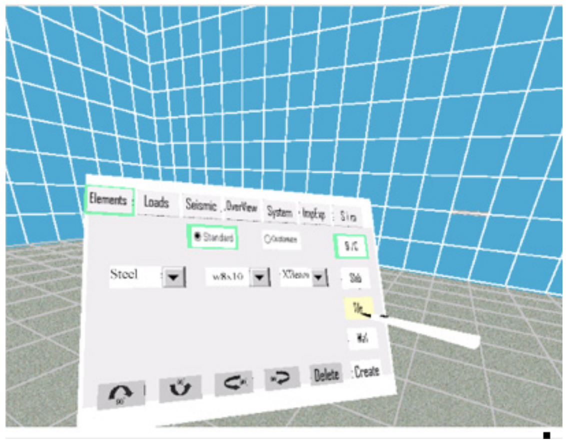


Figure 6.4. Virtual-SAP 1.0 user interface (Bowman et al. 2003)

6.4 Step 2: Generation of domain-specific interaction techniques

6.4.1 Description

The preceding definition in Chapter 3 also captures that the design of domain level interaction techniques can be conducted using conventional testbed approach because the explicit knowledge is well structured or defined and shared among applications to a significant degree. The three-level design framework is used for comparison purpose. We have proposed the design at the domain level as Step 2 of the interaction design process to enhance existing UE process. Note that this is not an absolute Step number 2 because the design of interaction techniques is iterative. We may have performed several loops of Step 1. If there are domain-specific interaction techniques available, they could have been integrated in Step 1.

This step is necessary because generic level interaction techniques do not handle specific cases and existing techniques at other levels might not be optimal for the current design task at hand. As suggested by cognitive fit theory (Vessey 1991), user's performance improves as the task presentation and problem representation match. For example, the sequence of cloning operations can be reformed in a taxonomical structure to make the interaction more efficient. Also, at the beginning of the Step 1, some functions or constraints may not be discovered that are suitable for VEs. It is often a parallel process requiring considerable trade-off studies.

There could be many ways to design at a domain-level depending on the type of domain knowledge captured. Knowledge has been classified into different categories.

Knowledge can be further used as an instrument to construct interaction techniques by following different path in the design framework, in a sense that the three-level framework serves as an instrument for design and comparison. An approach to compare interaction techniques we took and found useful was the taxonomy-based testbed design and evaluation approach following path 1->4->5->3. At this stage, designers can use testbed and vary parameters to create alternative interaction techniques. These techniques are independent of the specific application therefore support better reuse.

We applied this approach because taxonomy can represent the detailed design space and associated mechanisms of how users may interact with the system, allowing designers to vary and measure different factors in the taxonomy to generate alternative designs and options (Buxton 1983, Foley et al. 1984, Card et al. 1991, Jacob et al. 1994, Mine 1995, Bowman et al. 1999, Tan et al. 2001). The ability to generate alternative designs has been advocated in conventional (Simon 1981) and recent design methodologies (Cross 2000, Chen and Bowman 2006, Wolf et al. 2006). The taxonomy is further used to design a testbed in order to measure performance (Poupyrev et al. 1998, Bowman et al. 1999, Chen et al. 2004) (Figure 6.5). Guidelines of use are also written so that techniques can be easily chosen by practitioners who wish to apply the techniques to other applications.

We use taxonomy-based development approach also because it can enclose any domain knowledge in the process. For example, task relevant taxonomies define the action sequences or the metrics based on the physical parameters (Poupyrev et al. 1997, Bowman and Hodges 1999) or cognitive process and users (Fleishman and Quaintance 1984, Chen and Stanney 1999) that affect the performance or cognitive abilities (Fineberg 1995). Device relevant taxonomies (MacKenzie et al. 1991) was used to define the operation and therefore can be used to design new devices (Zhai and Senders 1997, Julier et al. 2002) or to suggest the use of certain devices (Jacob et al. 1994).

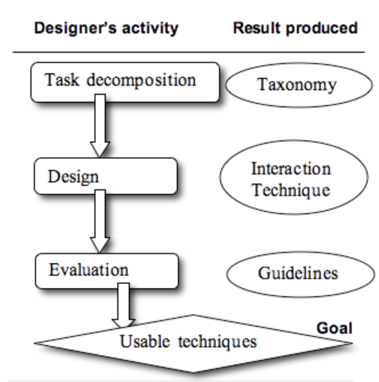


Figure 6.5. Testbed design and evaluation method

6.4.2 Example

We refine techniques designed for cloning given that it is the bottleneck found in the first usability studies. We further studied the possibility of designing cloning techniques based on the three-level design framework and how to make use of the techniques in the process (Chen and Bowman 2006). The set of new interaction techniques are produced from a cloning taxonomy. Rather than the reframing of the cloning by object selection and manipulation, the cloning operation was designed using a two-step action: users first select objects of interest and then generate new copies in the 3D space by defining numbers along each dimension (Chen et al. 2004).

Different styles of operation such as direct manipulation to indirect ones are considered. For example, an indirect way to perform cloning is to use sliders widget presented on a virtual tablet to define parameters. Users drag the sliders to specify the number of copies and the distance between copies on each axis. A physical tablet provides passive haptic feedback for better precision. This technique, while simple, still provides a much more powerful way to create many copies of objects in a 3D space.

Mostly importantly, the use of the three-level design framework became an instrument of discovering the domain knowledge use and reuse so that different design choices could be explored and made available. For example, adding spatial knowledge of repetitive pattern to HOMER forms the copy-by-example interaction technique. Considering users' mental knowledge of cloning actions form the keypad interface.

Design rationales and claims were written and the pros and cons of each were identified from usability studies. For example, the design rationale for a three-dimensional slider UI can be "the use of tablet is consistent to the Virtual-SAP application", "increasing the selection volume of the slider widgets makes selection easier", but "it is not easy to grab the widgets if they have more than two degrees-of-freedom or is floating in the tablet space". Also because of the well-constrained nature of the taxonomy, design rationale or claims produced from the testbed are easy to use.

Experience in the comparison of interaction techniques suggests that for cloning tasks, techniques directly addressing the domain task are more effective than those at the generic level in supporting better task performance and workflow (see Chapter 4). Results from the experiment also indicate that domain characteristics are especially useful for designing effective interaction techniques. All techniques designed with domain-level characteristics in mind outperformed those without for cloning tasks.

6.5 Integration Process

During the system design process, there is a need to balance dominant task characteristics and other factors to construct effective UIs so that different interaction techniques can function together. One way is to use a decision tree method (Cross 2000). A tree structure lists all possibilities of design choices and pro-and-cons are then marked on each node.

The decision tree often comes from task analyses where the task conditions are studied. Optimal solutions can be gained by comparing different routes. In this process, cross-task conditions that define the reuse of techniques can also be considered. For example, PORT (Lucas et al. 2005) can be used for cloning, objects selection for resizing selection box, and resizing beams and columns. Copy-by-example uses HOMER that can be reused for single object manipulation. HOMER can also be used for a target-based travel which is seemingly a useful technique for moving users quickly.

Integration process can place more constraints in the interface design. For example, the number of beams and columns can be represented by mathematical formulas that posses extra constraints to decide the number of beams and columns to be displayed. For example, the program should not draw the fourth beam if the user draws a building in which the horizontal span is four (which means four columns connected with three beams).

In the integration process, we chunk users' actions to allow more intuitive operations. For example, the interface allows users to create structures from scratch but not to rely on cloning existing structures. In a way, we keep users' tasks (e.g., building structure) in mind and designing the interface around them (e.g., a large button showing the shape of the structure but not purely cloning). In the integration process, we reuse PORT and copy-by-example as the candidate cloning techniques according to the results from comparing different cloning options.

6.6 Summary

Design is such a complex process that there is no one generic method for it. Following the idea of DSD, we can support interface design by means of producing domain-specific techniques that encode knowledge and prompt new interface design process. We reflected on our experiences in creating VE applications and presented a possible design process that deliberately addresses the design of effective interface through interaction. The top-down UE process and the DSD process are mutually supportive that would allow the design and comparison of interaction techniques in the UE process. The integrated process can have the potential to increase the usefulness and usability of applications.

CHAPTER 7: *Designing a Domain-Specific 3D UI for Architectural Massing Studies*

This chapter discusses how the integrated design approach is applied to the 3D UI design for massing study and what we have learned. We use the massing study as the second example of the integrated design process for at least two reasons. First, the integrated approach is discovered and analyzed after the Virtual-SAP is completed; this example provides a forward process to check if the process can be applied at least by ourselves. Though we did not go through the full design cycle, rather than testing a few examples, we present what we have learned and what other designers have done before us without this design process. Second, there was a large body of literature trying to deal with conceptual design without too much success. We like to test if the end users would be willing to try our UI for massing study resulted from DSD and combined design process.

7.1 Domain Analyses

The massing study is the phase that architects make mockups to examine their design ideas and to compare various design options. Requirement analyses were conducted with domain users in order to design a 3D UI for massing study. Conventional way of performing this work is to cut cardboard or shape clays to construct miniature models. If designers have different design options, they would build as many as needed to compare them. However, this process is time consuming given frequent changes and comparisons are often needed. It has been estimated that design decisions account for more than 75% of the final product cost (Hsu and Liu 2000). Material and machines cost climb up quickly as the number of models increases. Figure 7.1 illustrates the models used by architects to examine different layouts of a building for the design of Y2K in Rotterdam, the Netherlands (OMA 2000).

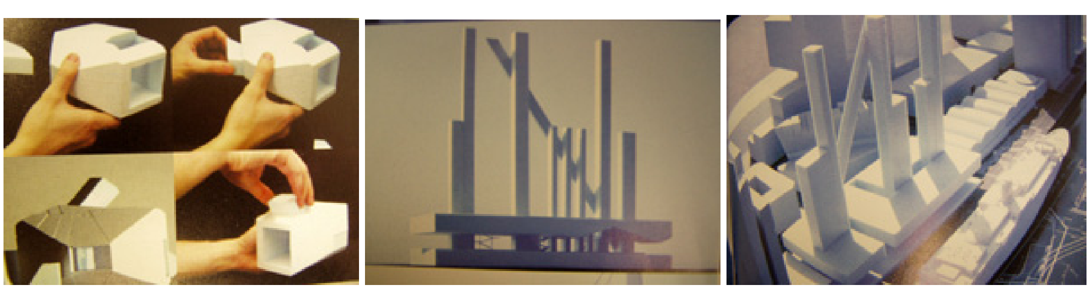


Figure 7.1. Models for massing study (Y2K)

One approach to increase efficiency in this process is to draw the model using desktop modeling tools and then send the drawings to a computer-aided manufacture (CAM) machine to cut the models. Designers then assemble different parts to form the final design. The problem is that sometimes the exact scales of the structure are unknown. Conventional desktop design tools have not been able to successfully support conceptual design that requires rough and implicit modeling, except probably for sketchUp®.

Early approaches using VEs for massing study (or conceptual design) have often been immersive modeling. Most existing approaches include complex menu systems that replicate desktop modeling tools (Sachs et al. 1991, Butterworth et al. 1992, Liang and Green 1994, Bowman 1996, Billinghurst et al. 1997, Mine 1997). There are dropdown menus, ring menus, menus on pen-and-tablet metaphor interfaces, and so forth. Architects can then pick basic shapes such as boxes and cylinders and place them in the 3D world. Colors, textures, and sizes can be edited through menu access as well. Some systems allow users to drag widgets to resize objects. There are also systems that are powered by rich functionalities. For example, curves and surfaces can be drawn, swept, or deformed (push in, add or remove material); the computer can then create surface models by fitting the surface parts (Kaber and Riley 2001).

Recently, systems are made more "intelligent" by using gesture recognitions (Zeleznik et al. 1996, Igarashi 1999, Schweikardt and Gross 2000, Vries and Achten 2002, Yoon et al. 2004). This type of intelligence is founded on the assumption that it is possible to capture and represent all structures knowledge and expert knowledge. Although there have been some successes, such systems can hardly be applied because of other tasks involved in this process and the ambiguity created. Drawing is not the sole goal. Checking the form, space, and aesthetics (e.g., symmetry, repetition, and instruction), checking how the building will fit in the surroundings, checking relative dimensions, and forming an image view impression are other tasks architects do at this stage that can hardly be represented by such intelligence. Algorithms, despite considerable research efforts, have yet to make significant inroad into the practice off design. Much more ambiguities need to be resolved by architects who use such systems (Yoon et al. 2004).

Tangible media is also applied (Ullmer and Ishii 1997, Underkoffler and Ishii 1999). For example, iSphere (Lee et al. 2006) is a dodecahedron ball installed with 12 sensors. It maps the hand actions of pull, push, and touch to the same operation on a model. However, it does not explain how to deal with multiple objects in the environments or opportunistic hand movements. Other theories and computational tools have been proposed on the basis of formal (mathematical) logic, analogies, cases, rules, constraints, and models. They somewhat have been designed with domain knowledge or characteristics in mind. For example, procedure modeling uses effective rule based logic to generate buildings or cities. However, this approach does not fit into the massing study phase by automating the shape creation.

In summary, most early existing work tried to mimic users' tasks of modeling geometrical shapes in 3D virtual environment. However, there has not been a great success in the simple treatment of complex massing design phase as drawing geometries.

On the contrary, most recent work replicates users' experience by direct drawing, but system usability is often limited by technology availabilities.

Despite all metaphor used, there is still no such environment that resembles the feeling of architects' office. An interaction or interface is good at handling one case but cannot deal with all actions using one device. For some immersive systems, immersion isolates the users from the physical world that architects usually enjoy staying in. Haptic devices are too intrusive to be placed and combined with immersive VEs for architects to use all day long.

7.2 Design Considerations for Domain Use

So can we design a system that architects can use on a daily basis? What are those characteristics lacking in the current VE setup? In this work, we focus on the use of designers' working context to benefit the user interface design. Rough object manipulation on simple shapes is a domain task for which we try to design interaction techniques, because this task is a fundamental requirement for massing studies.

Early focusing on the user and their design tasks suggests a rather simple approach in our current implementation. It is the design enhanced by *artifacts,* driven by the architects' understanding of the space, forms, and contexts. Rather than seeking to reduce or even eliminate the physical working space as the past approaches or involve too much intelligence in the design, the aim is to provide the architects with interactive design tools that allow them to design without considering the details but the form and structure of the designed structure.

Following the integrated UE process, we first design the overall massing study UI. The interface is used as a testbed to study interaction techniques to be produced. Using this interface and interaction techniques, architects can possibly *externalize* and *interpret* their design ideas and enhance their physical, mental, and sensory capabilities. The externalization can put their ideas to quick drawings whilst the interpretation put ideas to a test with different views.

According to the domain studies, architects form a personal design space with their surroundings. They use their space to gather, generate, compare, manipulate, and communicate information and knowledge related to various design concepts. In our design, we place a physical tabletop in the massing study environment to simulate architects' working context (Figure 7.2). The display devices could be a CAVE (cave automatic virtual environment) or a large screen display. Users' two hands are tracked with Polhemus magnetic trackers: left hand wearing a glove and right hand holding a stylus (a pen like device). With their two hands, they can stretch, rotate, scale, and grab objects directly on the tabletop.

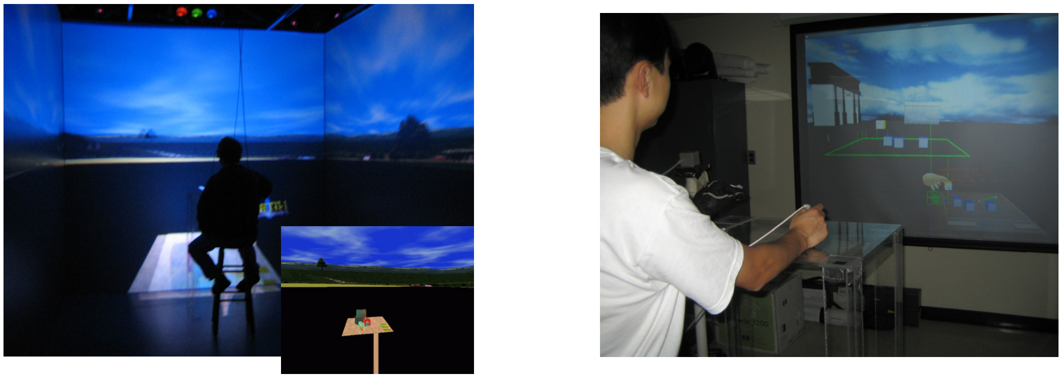


Figure 7.2. A hybrid virtual environment in a CAVE (left) or conventional office settings (right) for conceptual design

In this environment, users can remain seated or walk around. Two-handed tabletop interaction is designed to handle objects creation and modification. The system can also let the users examine Boolean operations and add physics for more realistic modeling of the space.

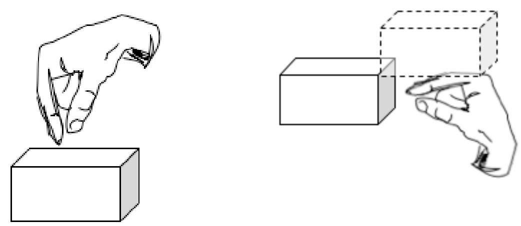
The ideas of using the table-prop are for ease of interaction and for domain consideration. First, architects work on their workbench everyday. The tabletop UI coupled with the two-handed input serves as designers' space to operate on virtual models. Second, the tabletop can be used as a manipulation and navigation aid and a secondary space to present information architects need. For example, for the CAVE setting, we project the images also on the table at a smaller scale; for the large screen display setting, the image of the table is projected onto the screen that coexists with the one to one scaled space. Third, the tabletop replicates architects' physical working environment, which is similar to the information kiosk (Anderson et al. 2003) where pictures of previous design can be picked. However, their system is fully immersed and is not dynamic in nature. Our system is flexible enough to be applied to conventional desktop or CAVE or HMD-based immersive VEs.

As a prototyping environment, we implemented one rough object modeling that includes object creation, manipulation, scaling, and rotation. Physical-based manipulation and Boolean operations are designed to aid the modeling operation. Using this interface, users can do rough placement and let the physics place the objects in place. Coupled with constraints provided by the table-prop, physics would allow the positioning in six degrees-of-freedom simultaneously easily (Froehlich et al. 2000).

The physical interaction between objects includes kinematics constraints, collision detection, and response. No full physics-based responses to external forces were implemented due to bouncing effects found from our pilot experiment which may decrease the speed and level of users' satisfaction. These physical behaviors were enough to afford the prediction of how objects move relative to each other.

In our first implementation, we turned on kinematics interactions and collision detection. Kinematics interactions involved one object's motion constraining or directly affecting the position and orientation of another object at some connection point or joint. And collision detection prevented objects from penetrating each other while positioned. The pilot study found that gravity helped lay objects on top of each other. However, users were frustrated because the existing structure could be toppled if a large object was picked up and accidentally hit the existing structure. This "accidental" hit may not happen in the physical world; however, it happens quite often in the virtual world possibly because virtual objects do not provide tactile feedback and they are virtual, hence do not convey their physical behavior. We therefore turned off this feature while an object was grabbed and turned it back on when the object was released. However, objects would penetrate into each other during maneuver in our current implementation.

Users using our system setup can perform the following interaction for rough objects manipulation (Figure 7.3). All actions are performed by pinching the index finger and the thumb or pressing down or releasing the stylus button.



(a). Move an object **(b). Copy an object**

Figure 7.3. Users interact with the objects

- Movement and duplication. The user points down then pinches two fingers just like we do in the physical world (+/- 45 degree); to copy an object (Figure 7.3 (b)), the user flats her hand to grab and move an object (Figure 7.3 (a)). The move and copy use the same gesture; only the orientation of the hand modifies the results.

- Deletion. The user grabs then drops it to a position out of the boundary of table on the side where the user is originally seated.

- Canvas movement. The user can hover over the canvas while pinching two fingers for about 4 seconds, the canvas will move following users' hand movement, similar to the real-world metaphor of moving a table cloth. This would allow users to move the current structure that is already built to some other places in the space for comparison purposes.

- Moving the canvas also avoids the problem of limited desk space because the user can utilize the space around.

- Scaling/rotation. The user pinches two fingers with the left hand and holds down the stylus button with the right hand simultaneously to stretch or squeeze.

- New object creation. Two “knapsacks” shown on the table allow users to create an object from scratch. Users can also grab any objects in the world to create a copy.

- Boolean. Two objects can be added and subtracted (difference and intersection).

The most similar operations to ours might be Bricks (Fitzmaurice et al. 1995). The Bricks interface is a graspable UI that allows direct control of virtual objects through physical bricks or control. However, Bricks are physical gadgets to be used to define user interaction. Gestures cannot be supported because the user’s hands are not tracked. Bricks have the flexibility of using the tangible gadgets to represent menus or objects, otherwise hardly visible in a system. The massing study user interface we designed could also be enhanced with tangible media if desired.

7.3 Discussion of the Domain-Specific Design

The design of the initial prototype of the massing user interface has followed the integrated usability engineering approach. The interface was designed assuming artifacts in architect’s workspace have major impact on how the architects think and act in space. Application contexts such as domain knowledge are taken into account to design the interface. The prototype we implemented is similar to application-specific level of design given the fact that the specific environment is set up for massing study use. At least one interaction technique was designed for each sub-task of rough object modeling task for massing study. Our argument is that the integrated design process may start from the application level design. The application itself becomes an instrument through which needs are captured.

The design of interaction techniques started from the generic level from which existing techniques were picked and integrated into the UI. The sub-tasks of modeling include selection, creation, deletion, scaling, rotation, and so forth. We used the virtual hand metaphor for objects selection, i.e., objects are touched in order to be selected. The creation of objects was originally designed as pen-based widget selection, and later added direct grabbing from the knapsacks for ease of selection. Because there were not suitable scaling and rotation techniques for the tabletop, we designed the two-handed pulling techniques to scale objects and a driver’s metaphor to rotate objects. The driver’s metaphor rotates the objects according to the relative direction of rotation of the two hands, similar to how architects rotate objects in the physical world. In a sense, the scaling and rotation techniques were designed specifically for this system setup and can be used/reused for any similar considerations.

Unlike previous massing study user interfaces built for immersive VEs, ours have several direct manipulation techniques guided by the design approach.

7.4 Evaluation

7.4.1 Method

The present study was designed with a concern for how people would encounter the tabletop VE we implemented. It serves an indirect goal of measuring the effects of design with artifacts characteristics. Would they interpret it as the way we design it? Can they design some structures? This is because we do not presume that all users will experience the system identically. The variations in understanding and in activity within the environment are of equal interest to use as any commonality that can be seen.

Similar to the way we evaluated Virtual-SAP, we use analyses drawn upon qualitative and interpretative tradition within the social sciences, rather than those that emphasize measurement and statistical analysis. We hope to gather data that is more descriptive and attempts to represent reality as experienced by participants. Therefore, the importance of viewing the meaning of experience and behavior in context and in its full complexity is uppermost. Participants' discourses were collected and interpreted.

Also because architects experience the world, but do not give explicit definition of what tasks to perform, our experiment matches what we should measure. We also hope to establish, through an example, the legitimacy of an alternative empirical approach and the theoretical orientation of measuring real world use.

7.4.2 Procedure

Eight architectural department students participated this study. They were between 25 and 41 years old. The display is a customized GeoWall®, i.e., a passive polarized projection-based system. It uses two Mitsubishi XD450U DLP projectors. Stereo view was not used so users did not wear glasses. The tracking system is the Polhemus magnetic tracking system. The user's left hand worn a pinch glove. All participants are right handed.

Participants were seated in front of the large screen at about 1.5 meters away, the closest point without blocking the images (Figure 7.4). They remained seated for the rest of the experiment. There are three sessions. The first session is a training session in which participants are given directions on the system setup and how to use the devices. Following this is the second session when participants are asked to explore the functions of the environments. They are told what functions are available but are not told how to do it. They are encouraged to comment on the UI. This session lasts as long as no more comments are given to the experimenter and the experimenter feels that the participants are comfortable using the UI. During the last session, participants are assigned three common structures to build (Figure 7.5).

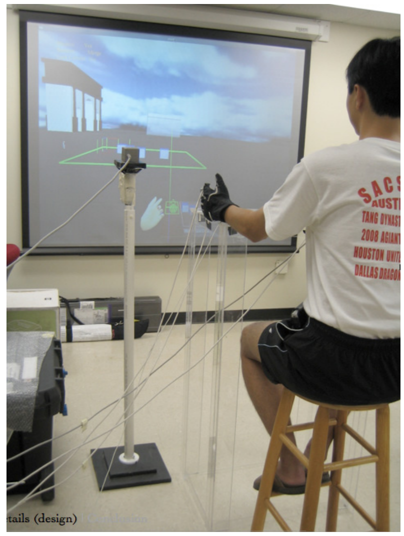


Figure 7.4. A user interacts in the massing study environment

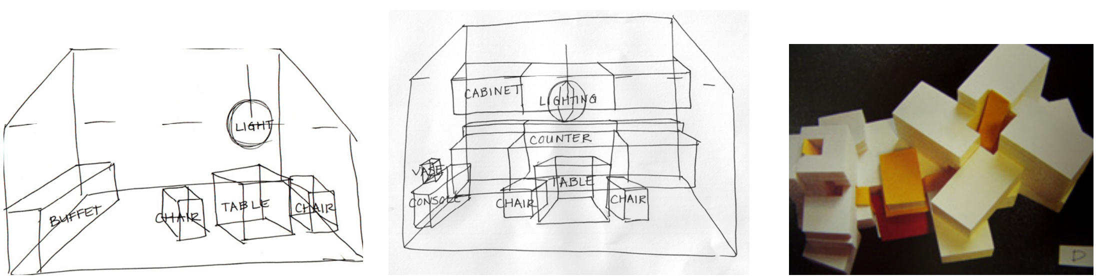


Figure 7.5. Three tasks for the massing study experiment

7.4.3 Major findings and discussions

The data analyses of participants' discourse and debrief interviews drew a number of interesting themes that had implications for the conceptual design. We did not explicitly ask about the domain knowledge use but indirectly gathered their comments about the use of tabletop and the physical setup. This would allow us to evaluate if our domain criteria were met.

In summary, participants liked the fact of integrating a tabletop in the environments. In fact, several participants immediately picked up the use and commented how they envisioned the modeling could have been helped with the use of the physical table that symbols the design space.

Table use. Participants generally were quite positive about the table use and seated condition, especially those participants who had VE experiences.

General architects: I imagine the table as a drafting table.

Interior designers: Can you put some physical blocks on the table because I view it as a room and the table surface as the floor?

Landscape architects: I would think this is the ground. (One of them later had trouble using the program because landscape architects deal with curves with different elevations.)

Most participants rest their arms on the table like what they normally do. Three participants complained that the table was too small to build large structures even if they could move the canvas to a virtual space. This would suggest that they made use of the table as a personal space to place objects. Also because the table was small, they could only work on a very fine detail but not move their hands at relatively large scales to scale or rotate objects.

All participants who used the CAVE before (four totally) commented that they liked the system to be set up in the CAVE. They also asked if I could display the world on the table so they did not have to look at the one on the screen. Most participants felt that the virtual table on the large screen display was too far from the physical table to link the action on the table to the display. A solution could be not to display the small virtual table at all on the large screen display since one participant also commented that he looked at the large world all the time since the mapping was more intuitive to him.

Related to the table use is the two-handed hand input. Users can grasp the concept fairly easily. As the answers in the post questionnaires indicated, the average rating of the objects creation is 5.5 (with 5 as somewhat easy and 6 as easy in a 1 to 7 scale, where 1 is very hard and 7 is very easy). The only problem is that the two knapsacks were placed on the left side of the table so the participants would use their left hands more if they just grab things from the knapsacks. Otherwise, they would have given a higher rating (The reason the knapsacks were on the left because the designer was ambidextrous). One participant said that was not a problem because the table was small. She reached the knapsacks with her right hand cross the table all the time.

Physics does not always help. It helped a lot when the participants operated on the table level so they could perform causal placement. The difficulty came when users tried to stack a group of small objects. Physics may not support some fancy structure. Also, one participant built buildings from top to bottom because he liked to visualize first the towers of high rise buildings. He turned off physics all the time.

Two participants commented that the scaling and rotation were easy to use with the table support because they could put their arms on the table to gain good control of the motion. Participants tried Boolean operations but the opinions were divided. Half of the participants said they always add stuff but never subtract because they “built” things. So even there was a hole in a mass, and even they had to cut a hole sometimes, they would prefer to form that mass by adding forms. Boolean operations should have a greater support on that. Currently, the Boolean operations assign equal weight on add, intersection, and difference.

Participants were asked the most frustrated factor while using the system. Four commented depth perception. Participants had a hard time judging the depth of the objects while they were placed, even when supported by visual cues. Although this was not against the idea of adding table to the environment, it did affect the choice of displays. While using CAVE, the user could lean to view the model displayed on the tabletop. While using the large screen display, the model was displayed further away at a distance that is far out of the human's ability to perceive the depth.

The tabletop was able to support target based travel very well similar to Bowman's target-based travel except that the travel is 3D. Participants did not have trouble understanding how travel worked and commented that the table defined a modeling space so they knew to where they were going.

7.5 Summary

In this chapter we prototyped a massing study system with the idea of integrating architects' working environments into VEs or bringing the tracking into architects' design environments. Because this is the very initial stage of the study, we were trying to learn if considering domain characteristics in constructing the system would offer any benefits and if the end users would like to use the system. Quite different from previous setup that tries to replicate the reality with digital modeling, our system also merges well with the reality, therefore augmenting those tasks performed on a computer.

We were able to use the three-level framework and the integrated UE process to quickly prototype a promising domain-specific 3D UI. Overall comments about using the tabletop interaction were positive. There are needs to do more quantitative studies to justify the benefits of using domain characteristics for system design. It is also important to answer: Do we need to design input/output devices for every application? Is there a way to build "domain-specific devices" so they can be reused at least within a domain if not for all applications? Also because it is in an early stage, we need to learn from the prototype to address domain-specific modeling. We propose this as future work.

CHAPTER 8: *Conclusion and Future Work*

This dissertation work was motivated by the lack of new categories of VE applications and the lack of highly interactive applications. The detailed analyses of related work suggested that the interaction techniques were too generic to meet real-world design requirements. This dissertation has attempted to address *what* and *how* to design interaction. A major research question was to answer what the effects of designing domain-specific interaction techniques (as opposed to general-purpose interaction techniques) are with regard to usefulness.

To attain this goal, we have developed a three-level design framework to make sense of interaction design and to insert the middle ground of domain-level design. The DSD approach is set out to utilize domain knowledge as a means to create *useful*, *usable*, and *re-usable* interaction techniques. DSD is not 3DUI-specific. For example, the cloning task is a domain task widely used in almost all desktop modeling tools. Since DSD does not limit the styles of interaction techniques, any desktop interaction techniques designed for the cloning task can be mapped to the three-level design framework. For example, AutoCAD uses command line input that follows the bottom up path; SketchUp directly address the domain task utilizing direct manipulation that follows the horizontal path in the three-level framework.

Domain knowledge use has been effectively used in 2D interaction design, but most is at a semantic level. For example, AutoCAD allows users to issue a "duplicate" (similar to clone) command that has a meaning to a domain. Also many commercial visualization tools (such as EVS, AVS) apply domain knowledge to visualization so that users can understand the UIs easily. EVS for underground water visualization allows engineers to visualize pollution conditions by layers. Such layers are well-defined in the environmental engineering domain.

We specifically apply DSD to 3D UIs in this dissertation because 3D interaction is much harder to design than 2D; there are few expert designers to design effective 3D interaction techniques.

Additionally, a combined design process integrating DSD and UE was used to design the overall 3D UIs. Based on our long-term research on the domain of AEC, the cloning task and the object manipulation task were used to demonstrate that the DSD approach is useful. The usefulness of the DSD approach was further demonstrated by applications.

The first example was to design interaction techniques for a cloning task. Following a three-level design framework and the DSD approach, we provided answers to the questions: Can we use domain knowledge to guide the design of interaction? How to produce the interaction techniques? Are there any benefits of designing at the domain level with regard to improving interaction techniques and 3D UIs? The research showed that the techniques we designed at the domain level were more effective than techniques we derived from the generic level. The techniques were further used in the Virtual-SAP application. Empirical evidence demonstrated that the new version of Virtual-SAP had a high level of usefulness compared to the previous version with only generic interaction techniques integrated. Of course, we have not proven that domain-specific techniques are always better than generic ones, nor that applications with domain-specific techniques are always more useful. Rather, we have demonstrated, through the design of many techniques and one application, that DSD is a reasonable and usable approach that can lead to high levels of effectiveness and usefulness for real-world tasks.

Our goal was not to compare DSD with other design approaches. Instead, we suggest that designers should also consider domain-level design of interaction techniques, and not simply the existing two levels. The goal was not to say that DSD is better but to say that adding domain knowledge benefits interaction.

Since we have gained a certain level of success while conducting the first phase, an integrated design process was proposed in order to integrate the DSD method into the UE process. A forward examination of the design process was carried out in the second phase. For this phase, the massing study demonstrated the use of the integrated design process with a prototype produced. The results indicate that people liked the system designed with domain characteristics in mind.

Tremendous research efforts were made on understanding design, i.e., the DSD approach and the three-level framework. A large body of data was produced pertaining to the performance of domain-specific interaction techniques for cloning in immersive virtual environments. These results demonstrate that it can be very effective to design domain-specific interaction techniques (as opposed to general-purpose interaction techniques) with regard to usefulness for domain-level tasks. In this research, all techniques that integrated domain knowledge outperformed generic ones, at least for the task of cloning. We have also produced the new Virtual-SAP application using the cloning interaction techniques. This provides extra evidence that DSD is a useful approach to building more useful 3D interfaces. Besides applying DSD to the cloning example, we present the use of device knowledge in designing 3D interaction for massing study.

Beyond these immediate results, our research has also developed a greater understanding of the design and evaluation of VE interaction and 3D applications. Here, we will briefly discuss several of these important contributions.

8.1 Domain-Specific Design Method

After Brooks visited major labs examining successful VE applications, he said that interacting more effectively with virtual world was one of the open challenges (Brooks 1999). By the year 1999, however, there was apparently not a lack of interaction techniques since the design boom started around 1995 (Augus and Sowizra 1995, Deering 1995, Mapes and Moshell 1995, Marcus 1995, Mine 1995, Poupyrev et al. 1996, Bowman et al. 1999). It seems that those successful applications did not make use of published results.

We may conclude that the low use of interaction techniques could have been caused by several reasons related to the gap between research and application. First, techniques were mainly designed at universities, so they may have been stuck in research labs. Second, designers may not have been aware of the interaction techniques. Third, the techniques may have been difficult to apply. The first two reasons seem unlikely to be true because those researchers were technological leads from top-notch research labs. And most interaction techniques were published in major conferences, such as VRST, VR, and UIST that should have been noticed by those researchers. This leaves the last reason, i.e., it was difficult to apply existing interaction techniques.

Most existing interaction techniques are designed for the generic-level tasks of navigation, manipulation, system control, and symbolic input. They are designed without consideration of any domain of use. However, all tasks take place within a context. Designing at the domain level to address real-world use makes more sense to designers and users. One major contribution of this work is therefore the interaction design approach called DSD. It has several advantages over conventional generic-level design. The design theory provides a foundation to tell what to design and how.

A theory-guided approach can also produce more robust results. Beyond DSD is a three-level design framework classified by domain specificity. The three-level framework can be used as an instrument to generate domain-level designs. Many design options can therefore be produced. Because there can be an exact mapping of the techniques to the framework, we can easily compare the pros and cons of each path through the framework.

Finally and most importantly, the interaction techniques produced from DSD can increase the usefulness of applications. We have shown through empirical studies that the new Virtual-SAP application had better usefulness in building large complex models. Designing at a domain level may reduce the overall cost of the application building process.

8.2 Integrated design process and evaluation approach

A second major contribution of this work is the integrated design process and evaluation approach. The method includes the integrated design process (of DSD and the conventional UE process) and an evaluation approach. The integrated design process was developed based on reflections on our experience in designing Virtual-SAP. One major advantage of using this integrated design process is to enable *design* but not pure *integration* (though this is ideal). For example, we can use the application as a testbed to measure the effectiveness of the interaction techniques. Many design options can be produced based on the three-level framework. Finally, the evaluation should discover what is unknown in addition to what is known. Our evaluation matrix includes many dimensions, such as performance measurement, social behavior and discourse, drawings, etc. With such measurements, we could gain a complete picture of usability.

8.3 Domain-Specific Guidelines

An ideal situation for practitioners would be that they could use interaction techniques like a module and synthesize them together with or without tweaking to form a useful VE. This is often realized by design rationales and guidelines from usability studies. In this dissertation work, many interaction techniques were designed for cloning and several for massing study. Here we list the guidelines of use so practitioners can decide what to use in the future.

Integrate domain knowledge whenever possible

A major take away lesson from our experiments is that domain characteristics play a significant role in the design process. Even if we only consider adding simple characteristics to interaction techniques, we can still have significant performance gains, as evidenced by the copy-by-example technique, modified from the HOMER interaction technique.

Produce interaction techniques by directly addressing the domain task if possible

Experimental data also suggest that workflow is better when techniques directly address the desired task. For example, PORT-C is better than copy-by-example for cloning because no tedious repetitive selection is required.

Use DSD and the three-level design framework to produce many design options

DSD and the three-level design framework define alternative ways to produce interaction techniques by adding domain knowledge. Practitioners or designers can produce many options by assigning different weight to alternative means.

Add domain knowledge according to the target users' needs, but more is not always better

In the continuum defined by the three-level design framework, some techniques are closer to the application level than others due to the number of domain constraints added. For example, the space-filling metaphor interface might be more constrained than PORT because it also defines the space. The space-filling technique, though strongly associated with the architectural domain, is limited in a way that it does not match exactly how engineers envision structural engineering. Architects are constrained by space, but engineers are often not.

Construct a taxonomy of domain knowledge for the target design task

Taxonomy has been applied effectively in conventional HCI design. It is quite effective for 3D interaction design as well. It has provided us a good understanding of the design space.

<u>*Build interaction techniques at the domain level*</u>

We feel that the best techniques are not those at the application level, but at the domain level because of their combination of reusability and specificity. The reusability was fulfilled by domain knowledge use. For example, cloning techniques can be applied in almost all 3D immersive modeling tools. Practitioners can pick up appropriate cloning techniques from our collection and directly apply them to specific applications. In this way, practitioners can tune the domain-level techniques to meet their application needs.

8.4 Future work

Several directions for future work are pointed out in this section. One of the dreams driving our work is to see the real-world use of VE technologies as the interaction design becomes more mature.

8.4.1 Domain knowledge library for reusable interaction

The success of DSD relies on domain knowledge use. The whole idea of using knowledge also lies in the reuse of interaction techniques. Knowledge has many kinds, but only a small piece was investigated thoroughly in this dissertation. One possible future work is to address the classification of domain knowledge and put it in a format that is easy to index and search, and is therefore reusable. The same can be done for the corresponding interaction techniques.

Reuse, however, is difficult because flexible adaptability is not readily available, particularly for subjects that are as difficult as interaction design. Interaction design is difficult not only because the tasks are complicated, but also because interaction design is so unconstrained as to lay open many possibilities. Certainly complexity may be addressed with hierarchical understanding and classification, usually employing some kind of parameterization.

We can possibly classify knowledge and investigate the tasks that can be framed. It is then possible to apply DSD and the three-level design framework to produce the interaction techniques. Since we have adopted Johnson's definition of domain knowledge (Johnson 1992), we can further classify knowledge into at least the following three categories:

- Semantic network. Networks of related facts, where the association may fall into several subtypes: information about objects in the scene, scenarios and claim to explain events or how things happen, information about stakeholders/users, and so on.

- Rules and heuristics knowledge. These include rules with IF-condition-THEN-action instructions, or less formal rules of thumb, such as heuristics; and general advice stated in natural language.

- Taxonomies and classifications. Taxonomic knowledge contains definitions for classes of action sequences, the class hierarchy structure, and the parameters of each class.

If we can have a collection of interaction techniques that address the knowledge at these levels, practitioners may apply them easily. The knowledge base can be connected to reuse as the development of the library progresses. For example, when we were at the massing study phase, the reusable components could only be scenarios or cases; after we have reached the cloning phase, the reusable components have expanded to taxonomies, rules, and semantic network at all levels. However there are limits to abstraction. As the design becomes complex, the dilemma of adaptability emerges. The more adaptable features an interaction technique has, the more of the designer's job is passed on to the practitioner. Therefore in each category, there could be hierarchies.

8.4.2 Applying DSD and the integrated process to other technologies

VEs provide an example of use in our work because 3D interaction was initiated from this field and has been developed for many years. We believe many nascent technologies, such as large-screen displays, ubiquitous computing, and augmented reality, should also be able to use DSD and the integrated design process.

The framework and the DSD approach can work as is. But the task space and the technique space may vary according to the form factors of different displays. Certain tasks might be more suitable for one technology than another.

8.4.3 Appreciating 3D interaction and understanding the effective matrix through usability studies

The success of a design theory needs to be tested through usability studies, which consist of iterative cycles of designing, prototyping, and evaluating. Usability evaluation in studying 3D UIs often covers a subset of the possible actions users take. Because of the difficulties in analyzing the data, many studies focus purely on the performance measurement of several main tasks. Users behaviors and action sequences are unknown. And variances of performance are left unexplained.

Interacting with different displays, using non-conventional input devices, and interacting using multiple platforms impose requirements beyond simply measuring users' task completion time. In 3D, individual problem-solving strategies, the underlying decision structure, and mental models of a given task have greater variability. Modeling of captured data from users' actions may support the interpretation of performance data, optimize training process, identify unknown problems, and further suggest new design solutions or the migration of problems.

The practical methods of empirical study, ranging in rigor from exploratory studies to hypothesis-testing experiments, provide vital bases for choosing interaction techniques, setting up systems, and examining the cause and effects. It has been widely accepted that different usability study approaches find different usability problems and that usability findings vary between evaluators even if the same evaluation technique is used. This would suggest that we could use multiple evaluators in the evaluation process, as suggested by Nielsen for heuristic evaluation (Nielsen 1993).

Approaches to automate usability analysis would standardize the measurements. By usability analysis, we refer to the process that automatically records and analyzes such data as actions and action sequences. We pick usability analysis because it is an error-prone process and because this process helps discover the true cause of a usability problem, if any.

References

Alben, L., "Defining the criteria for effective interaction design: quality of experience," *Interactions* (May.June), pp. 11-15, 1996.

Allen, B., "Individual differences and the conundrums of user centered design: two experiments," *Journal of the American Society for Information Science*, vol. 51 (6), pp. 508-520, 2000.

Anderson, L., Esser, J., & Interrante, V., "A Virtual Environment for Conceptual Design in Architecture," *Eurographics Workshop on Virtual Environments*, pp. 57-63, 2003.

Angus, I. and Sowizral, H., "Embedding the 2D interaction metaphor in a real 3D virtual environment," *Stereoscopic Displays and Virtual Reality Systems, 2409*, pp. 282-293, 1995.

Augus, I. and Sowizra, H., "Embedding the 2D interaction metaphor in a real 3D virtual environment," *SPIE (Stereoscopic Displays and Virtual Reality Systems)*, pp. 282-293, 1995.

Badler, N., Manoochehri, K., & Baraff, D., "Multi-dimensional Input Techniques and Articulated Figure Positioning by Multiple Constraints," *ACM Workshop on Interactive 3D Graphics*, pp. 151-170, 1986.

Balakrishnan, R., Fitzmaurice, G., Kurtenbach, G., & Singh, K., "Exploring Interactive Curve and Surface Manipulation Using a Bend and Twist Sensitive Input Strip," *Symposium on Interactive 3D Graphics*, pp. 111-118, 1999.

Bales, R.F., Interaction Process Analysis: a Method for the Study of Small Groups, 2nd ed. Chicago: University of Chicago Press, 1976.

Beaudouin-Lafon, M., "Instrumental Interaction: An Interaction Model for Designing Post-WIMP User Interfaces," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 446-453, 2000.

Billinghurst, M., "Put That Where? Voice and Gesture at the Graphics Interface," *Computer Graphics* 60-64, 1998.

Billinghurst, M., Baldis, S., Matheson, L., & Philips, M., "3D Palette: a Virtual Reality Content Creation Tool," *ACM Symposium on Virtual Reality Software and Technology*, pp. 155-156, 1997.

Bowman, D., Davis, E., Badre, A., & Hodges, L., "Maintaining Spatial Orientation during Travel in an Immersive Virtual Environment," *Presence: Teleoperators and Virtual Environments*, vol. 8 (6), pp. 618-631, 1999.

Bowman, D. and Hodges, L., "An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments," *Symposium on Interactive 3D Graphics*, pp. 35-38, 1997.

Bowman, D. and Hodges, L., "Formalizing the Design, Evaluation, and Application of Interaction Techniques for Immersive Virtual Environments.," *The Journal of Visual Languages and Computing*, vol. 10 (1), pp. 37-53, 1999.

Bowman, D., Johnson, D., & Hodges, L., "Testbed Evaluation of VE Interaction Techniques," *VRST'99*, pp. 26-33, 1999.

Bowman, D.A., "Conceptual Design Space - Beyond Walkthrough to Immersive Design," *Designing Digital Space*. New York: John Wiley & Sons, 1996, pp. 225-236.

Bowman, D.A., Gabbard, J.L., & Hix, D., "A Survey of Usability Evaluation in Virtual Environments: Classification and Comparison of Methods," *Presence: Teleoperators and Virtual Environments*, vol. 11 (4), pp. 402-424, 2002.

Bowman, D.A., Hodges, L.F., & Bolter, J., "The virtual venue: user-computer interaction in information-rich virtual environments," *Presence: teleoperators and virtual environments*, vol. 7 (5), pp. 478-493, 1998.

Bowman, D.A., Johnson, D.B., & Hodges, L.F., "Testbed Evaluation of Virtual Environment Interaction Techniques," *ACM Symposium on Virtual Reality Software and Technology*, pp. 26-33, 1999.

Bowman, D.A., Kruijff, E., LaViola, J.J., & Poupyrev, I., *3D User Interfaces: Theory and Practice*, 1st ed. Boston: Addison-Wesley, 2004.

Bowman, D.A., North, C., Chen, J., Polys, N., Pyla, P.S., & Yilmaz, U., "Information-Rich Virtual Environments: Theory, Tools, and Research Agenda," *Virtual Reality Software and Technology (VRST)*, pp. 2003.

Bowman, D.A., Setareh, M., Pinho, M.S., Ali, N., Kalita, A., Lee, Y., Lucas, J.F., Gracey, M., Kothapalli, M., Zhu, Q., Datey, A., & Tumati, P., "Virtual-SAP: an immersive tool for visualizing the response of building structure to environmental conditions," *IEEE Virtual Reality*, pp. 243-250, 2003.

Brooks, F., "Grasping Reality Through Illusion: Interactive Graphics Serving Science," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 1-11, 1988.

Brooks, F., Ouh-Young, J., Batter, J., & Kilpatrick, P., "Project GROPE- Haptic Displays for Scientific Visualization," in *SIGGRAPH'90*, vol. 24: ACM, 1990, pp. 177-185.

Brooks, F.P., "Walkthrough - a Dynamic Graphics System for Simulating Virtual Buildings," *Symposium on Interactive 3D Graphics*, pp. 9-21, 1986.

Brooks, F.P., "What's real about virtual reality," *IEEE Computer Graphics & Applications*, vol. Nov/Dec 16-27, 1999.

Bukowski, R. and Sequin, C., "Object Associations: A Simple and Practical Approach to Virtual 3D Manipulation," *Symposium on Interactive 3D Graphics*, pp. 131-138, 1995.

Burns, C., Bryant, D., & Chalmers, B., "Scenario mapping with work domain analysis," *Human factors and ergonomics society 45th annual meeting (HFES)*, pp. 424-428, 2001.

Burns, E., Razzaque, S., Whitton, M., McCallus, M., Panter, A., & Brooks, F., "The hand is slower than the eye: a quantitative exploration of visual dominance over proprioception," *IEEE Virtual Reality*, pp. 2005.

Butterworth, J., Davidson, A., Hench, S., & Olano, M.T., "3DM: a Three Dimensional Modeler Using a Head-Mounted Display," *Symposium on Interactive 3D Graphics*, pp. 135 - 138, 1992.

Button, G., "The ethnographic Tradition and Design," *Design Studies*, vol. 21 319-332, 2000.

Buxton, W., "Lexical and Pragmatic Consideration of Input Structures," *Computer Graphics*, vol. 17 (1), pp. 31-37, 1983.

Card, S.K., Mackinlay, J.D., & Robertson, G.G., "A Morphological Analysis of the Design Space of Input Devices," *ACM Transactions on Information Systems (TOIS)*, vol. 9 (2), pp. 99-122, 1991.

Carroll, J. and Rosson, M.B., "Getting Around the Task-Artefact Framework: How to Make Claims and Design by Scenario," *ACM Transactions of Information System*, vol. 10 181-212, 1992.

Carroll, J.M., Making Use: Scenario-Based Design of Human-Computer Interactions. Cambridge, MA: MIT Press, 2000.

Carroll, J.M., HCI Models, Theories and Frameworks: Toward a Multidisciplinary Science: Morgan Kaufmann Publishers, 2003.

Carroll, J.M. and Olson, J.R., "Mental Models in Human-Computer Interaction," in *Handbook of Human-Computer Interaction*. Amsterdam: Elsevier Science Publishers, 1988.

Chen, J. and Bowman, D.A., "Domain-specific design of interaction techniques: an approach for desiging useful virtual environment applications," *Submitted to the International Journal of Human-Computer Studies, also appear at* http://www.cs.vt.edu/~jichen8/publications/IJHCS/ChenBowman-DSDv1.pdf, 2006.

Chen, J. and Bowman, D.A., "Evaluation of the effectiveness of cloning techniques for architectural virtual environments," *IEEE Virtual Reality* pp. 2006.

Chen, J., Bowman, D.A., Lucas, J.F., & Wingrave, C.A., "Interfaces for cloning in immersive virtual environments," *Eurographics Symposium on Virtual Environments*, pp. 91-98, 2004.

Chen, J., Bowman, D.A., Wingrave, C.A., & Lucas, J.F., "Designing Explicit Numeric Input interfaces for Immersive Virtual Environments," Virginia Polytechnic and State University, Blacksburg Computer Science TR-04-13, 2004.

Chen, J. and Stanney, K.M., "A theoretical model of wayfinding in virtual environments: proposed strategies for navigational aiding," *Presence: Teleoperators and Virtual Environments*, vol. 8 (6), pp. 671-685, 1999.

Chen, M., Mountford, S.J., & Sellen, A., "A Study in Interactive 3D Rotation Using 2-D Control Devices," *Computer Graphics (ACM SIGGRAPH)*, vol. 22 (4), pp. 121-129, 1988.

Clark, H.H. and Brennan, S.E., "Grounding in Communication," in *Perspectives on Socially Shared Cognition*, L.B. Resnick, J. Levine, and S.D. Teaskey, Eds. Washington DC: American Psychological Association, 1991, pp. 127-149.

Cross, N., Engineering design methods: strategies for product design: Wiley, 2000.

Dachselt, R. and Hubner, A., "A survey and taxonomy," *Computer & Graphics*, vol. 31 (1), pp., 2007.

Davis, F.D., "Perceived usefulness, perceived ease of use, and user acceptance of information technology," *MIS Quarterly*, vol. 13 319-340, 1989.

Deering, M.F., "HoloSketch: a Virtual Reality Sketching/Animation Tool," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 2 (3), pp. 220-238, 1995.

Diaper, D., Task Analysis for Human-Computer Interaction. New York: Wiley, 1989.

Dix, A., Finlay, J., Abowd, G., & Beale, R., *Human-Computer Interaction*: The university Press, Cambridge, 1993.

Durlach, N.I. and Mavor, A.S., *Virtual Reality: Scientific and Technological Challenges*. Washington D.C.: National Academy Press, 1995.

Ekstrom, R.B., French, J.W., & Harman, H.H., *Manual for kit of factor referenced cognitive tests*. Princeton, NJ: Educational testing service, 1976.

Feiner, S. and MacIntyre, B., "Knowledge-Based Augmented Reality," *Communications of the ACM*, vol. 36 (7), pp. 53-61, 1993.

Fineberg, M., "A comprehensive taxonomy of human behavior for synthetic forces," *IDA paper N. P-3155. VA: Institute for defense analyses*, 1995.

Fischer, G., "Beyond human-computer interaction: designing useful and usability computational environments," in *People and Computers*, vol. VIII, J. Alty, D. Diaper, and S. Guest, Eds. Cambridge, UK: Cambridge University Press, 1993.

Fischer, G., "Articulating the Task at Hand and Making Information Relevant to It," *Human-Computer Interaction*, vol. 16, 2001.

Fitts, P.M., "The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement," *Journal of Experimental Psychology*, vol. 47 381-391, 1954.

Fitzmaurice, G.W., Ishii, H., & Buxton, W., "Bricks: Laying the Foundations for Graspable User Interfaces," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 442-449, 1995.

Fleishman, E.A. and Quaintance, M.K., *Taxonomies of human performance: the description of human tasks*. New York: Academic Press, 1984.

Foley, J.D., Wallace, V.L., & Chan, P., "The Human Factors of Computer Graphics Interaction Techniques," *IEEE Computer Graphics and Applications*, vol. 4 13-48, 1984.

Forsyth, D.A. and Ponce, J., *Computer Vision: A Modern Approach*, 1st ed: Prentice Hall, 2002.

Forsyth, D.R., *Group Dynamics*, 3rd ed. Belmont, CA: Wadsworth Publishing Company, 1999.

Frees, S. and Kessler, G., "Precise and Rapid Interaction Through Scale Manipulation in Immersive Virtual Environments," *IEEE VR*, pp. 99-106, 2005.

Froehlich, B., Tramberend, H., Beers, A., Agrawala, M., & Baraff, D., "Physically-Based Manipulation on the Responsive Workbench," *IEEE VR*, pp. 5-11, 2000.

Fuchs, H., Goldfeather, J., Hulquist, J.P., Spach, S., Austin, J.D., Brooks, F.P.J., Eyles, J.G., & Poulton, J., "Fast spheres, shadows, textures, transparencies, and image enhancement in pixel planes," *Computer graphics*, vol. 19 (3), pp. 111-120, 1985.

Gabbard, J.L., Hix, D., II, J.E.S., Livingston, M.A., Hollerer, T.H., Julier, S.J., Brown, D., & Baillot, Y., "Usability Engineering for Complex Interactive Systems Development," *Human Systems Integration Symposium 2003, Engineering for Usability*, pp. 1-13, 2003.

Garfinkel, H. and Sacks, H., "The Formal Practices of Practical Actions," in *Theoretical Sociology*, J.C. McKinney and E.A. Tiryakian, Eds. New York: Appleton-century-Crofts, 1970, pp. 338-366.

Gleicher, M. and Witkin, A., "Through-the-Lens Camera Control," *Computer Graphics (SIGGRAPH'92)*, vol. 26 (2), pp. 331-340, 1992.

Goesele, M. and Stuerzlinger, W., "Semantic Constraints for Scene Manipulation," *Spring Conference in Computer Graphics '99*, pp. 140-146, 1999.

Gordon, S.E., Systematic Traning Program Design: Maximizing Effectiveness and Minimizing Liability. Englewood Cliffs, NJ: Prentice-Hall, 1994.

Gordon, S.E., Schmierer, K.A., & Gill, R.T., "Conceptual Graph Analysis: Knowledge Acquisition for Instructional System Design," *Human Factors*, vol. 35 459-481, 1993.

Gould, J.D. and Lewis, C., "Designing for Usability: Key Principles and What Designers Think," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 1983.

Green, M. and Jacob, R., "Software Architectures and Metaphors for Non-WIMP User Interface," *Computer Graphics (ACM SIGGRAPH)*, vol. 25 (3), pp. 229-235, 1990.

Green, T.R.G., "Cognitive Dimensions of Notations," in *People and Computers V*, A. Sutcliffe and L. Macaulay, Eds. Cambridge, UK, 1989, pp. 443-460.

Grossman, T., Balakrishnan, R., Kurtenbach, G., Fitzmaurice, G.W., Khan, A., & Buxton, B., "Creating Principal 3D Curves with Digital Tape Drawing," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 121-128, 2002.

Grossman, T., Balakrishnan, R., & Singh, K., "An Interface for Creating and Manipulating Curves Using a High Degree-of-Freedom Curve Input Device," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 185-192, 2003.

Gulliksen, J. and Sandblad, B., "Domain-specific design of user interfaces," *International Journal of Human-Computer Interaction*, vol. 7 (2), pp. 135-151, 1995.

Heller, D., Ferguson, P.M., & Brennan, D., *Motif programming manual*: O'Reilly & Associates, 1994.

Hinckley, K., Pausch, R., Goble, J., & Kassell, N., "A survey of design issues in spatial input," *UIST `94*, pp. 213-22, 1994.

Hinckley, K., Pausch, R., Goble, J., & Kassell, N.F., "Passive Real-World Interface Props for Neurosurgical Visualization," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 452-458, 1994.

Hix, D. and Gabbard, J.L., "Usability engineering of virtual environments," in *Handbook of Virtual Environments*, K. Stanney, Ed. Mahwah, NJ: Lawrence Erlbaum Associates, 2002, pp. 681-700.

Hix, D. and Hartson, H.R., Developing User Interfaces: Ensuring Usability Through Product & Process: New York: John Wiley & Sons, 1993.

Hix, D., Swan, J.E.I., Gabbard, J.L., McGee, M., Durbiin, J., & King, T., "User-Centered Design and Evaluation of a Real-Time Battlefield Visualization Virtual Environment," *IEEE Virtual Reality*, pp. 96-103, 1999.

Holt, P.O.B., Ritchie, J.M., Day, P.N., Simmons, J.E.L., Robinson, G., Russell, G.T., & Ng, F.M., "Immersive Virtual Reality In Cable and Pipe Routing: Design Metaphors and Cognitive Ergonomics," *Computing and Information Science in Engineering*, vol. 4 (3), pp. 161-170, 2004.

Hornbak, K. and Frokjar, E., "Comparing Usability Problems and Redesign Proposals as Input to Practical Systems Development," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 391-400, 2005.

Hsu, W. and Liu, B., "Conceptual Design: Issues and Challenges," *Computer-Aided Design*, vol. 32 (14), pp. 849-850, 2000.

Hutchins, E.L., Hollan, J.D., & Norman, D.A., "Direct manipulation interfaces," *Human-computer interaction*, vol. 1 (4), pp. 311-338, 1985.

Igarashi, T., "Freeform User Interfaces for Graphical Computing," in *Computer Science*: The University of Tokyo, 1999.

ISO/IEC9241-14, "Ergonomic requirements for office work with visual display terminals (VDT)s - Part 14 menu dialogue," 1998.

Jacob, R., Siber, L.E., McFarlane, D.C., & Muller, M.P., "Integrality and Separability of Input Devices," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 1 (1), pp. 3-26, 1994.

Jacob, R.J.K., Siber, L.E., McFarlane, D.C., & Muller, M.P.J., "Integrality and Separability of Input Devices," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 1 (1), pp. 3-26, 1994.

Johnson, P., Human Computer Interaction: Psychology, Task Analysis and Software Engineering. New York: McGraw-Hill, 1992.

Julier, S., Baillot, Y., Brown, D., & Lanzagorta, M., "Information Filtering for Mobile Augmented Reality," *IEEE Computer Graphics and Applications* 12-15, 2002.

Kaber, D.B. and Riley, J.M., "Virtual Reality for Scientific Data Visualisation," in *Engineering Psychology and Cognitive Ergonomics (Volumn six): Industrial Ergonomics, HCI and Applied Cognitive Psychology*, D. Harris, Ed.: Ashgate, 2001, pp. 151-158.

Kalawsky, R.S., "New Methodologies and Techniques for Evaluating user Performance in Advanced 3D Virtual interfaces " *IEE, the 3D User Interface for the Information Worker* (May), pp., 1998.

Kalawsky, R.S., "VRUSE - a Computerised Diagnostic Tool: for Usability Evaluation of Virtual / Synthetic Environment Systems," *Applied Ergonomics*, vol. 30 11-25, 1999.

Keefe, D., Acevedo, D., Moscovich, T., Laidlaw, D., & LaViola, J., "CavePainting: A Fully Immersive 3D Artistic Medium and Interactive Experience," *Symposium on Interactive 3D Graphics*, pp. 85-93, 2001.

Kelso, J., Arsenault, L., Kriz, R., & Satterfield, S., "DIVERSE: A Framework for Building Extensible and Reconfigurable Device Independent Virtual Environments," *IEEE Virtual Reality*, pp. 183-190, 2002.

Kessler, G., Bowman, D.A., & Hodges, L.F., "The Simple Virtual Environment Library: an Extensible Framework for Building VE Applications," *Presence: Teleoperators and Virtual Environments*, vol. 9 (2), pp. 187-208, 2000.

Kirwan, B. and Ainsworth, L.K., *A Guide to Task Analysis*: Taylor and Francis, 1992.

Kitamura, Y., Ogata, S., & Kishino, F., "A Manipulation Environment of Virtual and Real Objects Using a Magnetic Metaphor," *Virtual Reality System and Technology*, pp. 2002.

Kitamura, Y., Yee, A., & Kishino, F., "A Sophisticated Manipulation Aid in a Virtual Environment using Dynamic Constraints among Object Faces," *Presence: Teleoperators and Virtual Environments*, vol. 7 (5), pp. 460-477, 1998.

Kohli, L. and Whitton, M., "The haptic hand: providing user interface feedback with the non-dominant hand in virtual environments," *Graphics interface*, pp. 2005.

Kopper, R., Ni, T., Bowman, D., & Pinho, M., "Design and evaluation of navigation techniques for multiscale virtual environments," *Virtual reality*, pp. 2006.

Krasner, G.E. and Pope, S.T., "A description of the model-view-controller user interface paradigm in the Samlltalk80 system," *Journal of Object Oriented Programming*, vol. 1 (3), pp. 26-49, 1988.

Kuester, F. and Hutchinson, T.C., A virtualized laboratory for earthquake engineering education: Wiley Press, 2005.

Kurmann, D., "Sculptor - how to design space?," *CAADRIA ' 98*, pp. 317-325, 1998.

LaViola, J.J., "Whole-Hand and Speech Input in Virtual Environments," in *Computer Science*: Florida Atlantic University, 1996.

LaViola, J.J., Feliz, D.A., & Keefe, D.F., "Hands-Free Multi-scale Navigation in Virtual Environments," *Symposium on Interactive 3D Graphics*, pp. 9-15, 2001.

Lee, C.H., Hu, Y., & Selker, T., "iSphere: a free-hand 3D modeling interface," *International journal of Architectural Computing*, vol. 4 (1), pp. 19-31, 2006.

Leganchuk, A., Zhai, S., & Buxton, W., "Manual and Cognitive Benefits of two-Handed Input: an Experimental Study," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 5 (4), pp. 326-359, 2000.

Liang, J. and Green, M., "JDCAD: A Highly Interactive 3D Modeling System," *Computers Graphics*, vol. 18 (4), pp. 499-506, 1994.

Lin, C.-Y., Chen, D.T., Loftin, R.B., Chen, J., & Leiss, E.L., "Interacting with visible human data using an ImmersaDesk," *IEEE Virtual Reality*, pp. 267-268, 2002.

Lindeman, R.W., Sibert, J.L., & Hahn, J.K., "Hand-held windows: towards effective 2D interaction in immersive virtual environments," *IEEE Virtual Reality*, pp. 205-212, 1999.

Loftin, R.B. and Kenney, P., "Training the Hubble Space Telescope Flight Team," *IEEE Computer Graphics & Applications*, vol. 15 (5), pp. 31-37, 1995.

Lok, B., Naik, S., Whitton, M., & Brooks, F.P.J., "Incorporating Dynamic Real Objects into Immersive Virtual Environments," *Symposium on Interactive 3D Graphics*, pp. 31-40, 2003.

Lucas, J.F., Bowman, D.A., Chen, J., & Wingrave, C.A., "Design and evaluation of 3D multiple object selection techniques," *Symposium on Interactive 3D Graphics*, pp. 2005.

Lucas, J.F., Kim, J., & Bowman, D.A., "Resizing beyond widgets: object resizing techniques for immersive virtual environments," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 1601-1604, 2005.

MacKenzie, I.S., Sellen, A., & Buxton, W., "A Comparison of Input Devices in Elemental Pointing and Dragging Tasks," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 161-166, 1991.

MacKenzie, I.S. and Ware, C., "Lag as a Determinant of Human Performance in Interactive Systems," *Interchi*, pp. 488-493, 1993.

Maiden, N.A. and Sutcliffe, A.G., "Exploiting Reusable Specifications Through Analogy," *Communication of ACM*, vol. 33 55-64, 1992.

Mapes, D. and Moshell, J., "A Two-Handed Interface for Object Manipulation in Virtual Environments," *Presence: Teleoperators and Virtual Environments*, vol. 4 (4), pp. 403-416, 1995.

Marcus, A., "Metaphor Design in User Interfaces: How to Effectively Manage Expectation, Surprise, Comprehension, and Delight," *Mosaic of Creativity, ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 1-2, 1995.

Marshall, R., Wilson, R., & Carlson, W., "Procedure models for generating three-dimensional terrain," *SIGGRAPH* 154-162, 1980.

McGuffin, M., Tancau, L., & Balakrishnan, R., "Using deformations for browsing volumetric data," *IEEE Visualization*, pp. 401-408, 2003.

Mine, M., "Virtual environment interaction techniques," UNC Chapel Hill CS Dept., Technical Report TR95-018, 1995.

Mine, M., "ISAAC: A Meta-CAD System for Virtual Environments," *Computer-Aided Design*, vol. 29 (8), pp. 547-553, 1997.

Mine, M., Brooks, F., & Sequin, C., "Moving objects in space: exploiting proprioception in virtual-environment interaction," *SIGGRAPH'97*, pp. 19-26, 1997.

Mine, M.R., "Virtual environment interaction techniques," Computer Science Department, University of North Carolina TR95-018, 1995.

Mine, M.R., "A Meta-CAD System for Virtual Environment," *Computer-Aided Design*, vol. 29 (8), pp. 547-553, 1997.

Mine, M.R., "ISAAC: A Meta-CAD System for Virtual Environments," *Computer-Aided Design*, vol. 29 (8), pp. 547-553, 1997.

Moran, T.P., "Getting Into a System: External-Internal Task Mapping," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 45-49, 1983.

Mosegaard, J., Ballisager, D., Carstensen, B., & Rasmusson, A., "Virtual open heart surgery: training complex surgical procedures in congenital heart disease," *Emerging technologies sketch, SIGGRAPH*, pp. 2006.

Muller, A., Conrad, S., & Kruijff, E., "Multifaceted Interaction with a Virtual Engineering Environment using a Scenegraph-oriented Approach," *International Conferences in Central Europe on Computer Graphics, Visualization and Computer Vision*, pp. 2003.

Muller, P., Wonka, P., Haegler, S., Ulmer, A., & Gool, L.V., "Procedure modeling of buildings," *SIGGRAPH*, pp. 2006.

Myers, B.A., "A new model for handling input," *Transactions on information systems*, vol. 8 (3), pp. 289-320, 1990.

Nardi, B.A., Context and Consciousness: activity theory and human-computer interaction. Cambridge, Mass.: MIT Press, 1996.

Nichols, J., Myers, B.A., & Litwack, K., "Improving automatic interface generation with smart templates," *Intelligent user interfaces*, pp. 286-288, 2004.

Nielsen, J., "Noncommand user interfaces," *Communication of ACM*, vol. 36 83-99, 1993.

Nielsen, J., *Usability Engineering*. New York: AP Professional, 1993.

Norman, D.A., "Cognitive Engineering," in *User Centered System Design*, D.A. Norman and S.D. Draper, Eds.: Hillsdale, NJ: Lawrence Erlbaum Associates, 1986, pp. 31-61.

Oh, J.-Y. and Stuerzlinger, W., "Moving objects with 2D input devices in CAD systems and desktop virtual environments," *Graphics interface*, pp. 2005.

Oh, J.-Y., Stuerzlinger, W., & Danahy, J., "SESAME: towards better 3D conceptual design systems," *ACM Designing Interaction Systems (DIS)*, pp. 80-89, 2006.

Olwal, A. and Feiner, S., "The Flexible Pointer: An Interaction Technique for Selection in Augmented and Virtual Reality," *ACM Symposium on User Interface Software and Technology*, pp. 81-82, 2003.

OMA, "Special issue: works/projects: Y2K," in *Architecture and Urbanism*, 2000.

Pair, J., Allen, B., Dautricourt, M., Treskunov, A., Liewer, M., Graap, K., Reger, G., & Rizzo, A.S., "A virtual reality exposure therapy application for Iraq war post traumatic stress disorder," *Virtual Reality*, pp. 67-72, 2006.

Parish, Y.I.H. and Muller, P., "Procedural Modeling of Cities," *SIGGRAPH*, pp. 301-308, 2001.

Pausch, R., Burnette, T., Brockway, D., & Weiblen, M., "Navigation and Locomotion in Virtual Worlds via Flight into Hand-Held Miniatures," *SIGGRAPH'95, Technical Sketches*, pp. 399-400, 1995.

Piekarski, W. and Thomas, B.H., "Interactive Augmented Reality Techniques for Construction at a Distance of 3D Geometry," *Eurographics Workshop on Virtual Environments*, pp. 19-28, 2003.

Pizer, S.M., Fuchs, H., Mosher, C., Lifshitz, L., Abram, G.D., Ramanathan, S., Whitney, B.T., Rosenman, J.G., Stabb, E.V., Chaney, E.L., & Sherouse, G., "3D shaded graphics in radiotherapy and diagnostic imaging," *Computer graphics*, pp. 107-113, 1986.

Poupyrev, I., Billinghurst, M., Weghorst, S., & Ichikawa, T., "The Go-Go Interaction Technique: non-linear Mapping for Direct Manipulation in VR," *ACM User Interface System And Technology (UIST)*, pp. 79-80, 1996.

Poupyrev, I., Tomokazu, N., & Weghorst, S., "Virtual Notepad: Handwriting in Immersive VR," *VRAIS*, pp. 126-132, 1998.

Poupyrev, I., Weghorst, S., Billinghurst, M., & Ichikawa, T., "A framework and testbed for studying manipulation techniques for immersive VR," *Virtual Reality System And Technology (VRST)*, pp. 21-28, 1997.

Poupyrev, I., Weghorst, S., Billinghurst, M., & Ichikawa, T., "Egocentric object manipulation in virtual environments: empirical evaluation of interaction techniques," *Computer Graphics Forum, EUROGRAPHICS'98 issue*, vol. 17 (3), pp. 41-52, 1998.

Rasmussen, J., Pejtersen, A.M., & Goodstein, L., *Cognitive systems engineering*. New York: Wiley, 1994.

Rasmussen, J. and Vicente, K., "Ecological interfaces: a technical imperative in hightech systems?" *International journal of human-computer interaction*, vol. 2 (2), pp. 93-111, 1990.

Regenbrecht, H., Kruijff, E., Donath, D., Seichter, H., & Beetz, J., "VRAM - a Virtual Reality Aided Modeler," *ECAADE*, pp. 235-237, 2000.

Robinett, W. and Rolland, J.P., "A computational model for the stereoscopic optics of a head-mounted display," *Presence: teleoperators and virtual environments*, vol. 1 (1), pp. 45-62, 1992.

Rosson, M.B. and Carroll, J., Usability Engineering: Scenario based development of Human-Computer Interaction. NY: Morgan Kaufmann, 2002.

Ruddle, R.A., Savage, J.C.D., & Jones, D.M., "Evaluating Rules of Interaction for Object Manipulation in Cluttered Virtual Environments," *Presence: Teleoperators and Virtual Environments*, vol. 11 (6), pp. 591-609, 2002.

Sachs, E., Roberts, A., & Stoops, D., "3-Draw: a Tool for Designing 3D Shapes," *IEEE Computer Graphics and Applications*, vol. 11 (6), pp. 18-26, 1991.

Sanderson, P.M., "Cognitive work analysis," in *HCI models, theories, and frameworks toward a multidisciplinary science*, J.M. Carroll, Ed.: Morgan Kaufmann Publishers, 2003, pp. 225-264.

Schkolne, S., Pruett, M., & Schroder, P., "Surface Drawing: Creating Organic 3D Shapes with the Hand and Tangible Tools," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 261-268, 2001.

Schon, D.A., *The reflective practitioner*: Basic book, 1983.

Schweikardt, E. and Gross, M.D., "Digital Clay: Deriving Digital Models from Freehand Sketches," *Automation in Construction*, vol. 9 107-115, 2000.

Setareh, M., Bowman, D.A., & Kalita, A., "Development of a Virtual Reality Structural Analysis System," *Journal of Architectural Engineering*, vol. 11 (4), pp. 156-164, 2005.

Setareh, M., Bowman, D.A., & Tumati, P., "Development of a Collaborative Design Tool for Structural Analysis in an Immersive Virtual Environments," *International Building Performance Simulation Associate Conference*, pp. 2001.

Shaw, C.D. and Green, M., "THRED: A Two-Handed Design System," *Multimedia Systems*, vol. 5 (2), pp. 126-139, 1997.

Shepard, R.N. and Metzler, J., "Mental rotation of three dimensional objects," *Science*, vol. 171 701-703, 1971.

Shneiderman, B., Designing the User Interface: Strategies for Effective Human-Computer Interaction, 3rd ed: Addison Wesley Longman, Inc., 1998.

SIGCHI, "ACM SIGCHI curricula for human-computer interaction," ACM Special Interest Group on Computer-Human Interaction Curriculum Development Group 1992.

Simon, A., *The sciences of the artificial*. Cambridge, MA: The MIT press, 1981.

Sloan, P.-P.J., Rose, C.F.I., & Cohen, M.F., "Shape by Example," *I3D*, pp. 135-143, 2001.

Stanney, K., "Realizing the full potential of virtual reality: human factors issues that could stand in the way," *VRAIS'95*, pp. 28-34, 1995.

Stanney, K.M., Mollaghasemi, M., Reeves, L., Breaux, R., & Graeber, D.A., "Usability Engineering of Virtual Environments (VEs): Identifying Multiple Criteria that Drive Effective VE System Design," *Int. J. Human-Computer Studies*, vol. 58 447-481, 2003.

Stanney, K.M., Mourant, R.R., & Kennedy, R.S., "Human factors issues in virtual environments: a review of the literature," *Presence: Teleoperators and Virtual Environments*, vol. 7 (4), pp. 327-351, 1998.

Stoakley, R., Conway, M., & Pausch, R., "Virtual reality on a WIM: interactive worlds in miniature," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 265-272, 1995.

Stuerzlinger, W. and Smith, G., "Efficient manipulation of object groups in virtual environments," *IEEE VR*, pp. 251-258, 2002.

Sutcliffe, A., The domain theory: patterns for knowledge and software reuse: L. Erlbaum Associates, 2002.

Sutcliffe, A., User-Centred Requirement Engineering: Theory and Practice. London: Springer Verlag, 2002.

Sutcliffe, A., Multimedia and Virtual Reality: Designing Multisensory User Interfaces: Lawrence Erlbaum Associates, Inc., 2003.

Sutcliffe, A.G., Benyon, D., & van Assche, F., *Domain knowledge for interactive system design*: Chapman & Hall, 1996.

Sutcliffe, A.G. and Carroll, J.M., "Designing Claims for Reuse in Interactive Systems Design," *International Journal of Human-Computer Studies*, vol. 50 213-241, 1999.

Sutcliffe, A.G., Kurniawan, S., & Shin, J.-E., "A Method and Advisor Tool for Multimedia User Interface Design," *Int. J. Human-Computer Studies*, vol. 64 375-392, 2006.

Sutherland, I., "The Ultimate Display," *IFIP Congress*, pp. 505-508, 1965.

Szalavari, Z. and Gervautz, M., "The Personal Interaction Panel - a Two-Handed Interface for Augmented Reality," *Computer Graphics Forum*, vol. 16 (3), pp. 335-346, 1997.

Szekely, P., Luo, P., & Neches, R., "Beyond interface builders: model-based interface tools," *INTERCHI*, pp. 383-390, 1993.

Tan, D.S., Robertson, G.G., & Czerwinski, M., "Exploring 3D Navigation: Combining Speed-coupled Flying with Orbiting," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 418-425, 2001.

Ullmer, B. and Ishii, H., "The metaDESK: Models and Prototypes for Tangible User Interfaces," *UIST*, pp. 223-232, 1997.

Underkoffler, J. and Ishii, H., "Urp: a luminous-tangible workbench for urban planning and design," *ACM conference on human factors in computing systems (CHI)*, pp. 386-393, 1999.

Vessey, I., "Cognitive fit: an empirical study of information acquisition," *Information system research*, vol. 2 63-84, 1991.

Vicente, K., *Cognitive work analysis*. Mahwah, NJ: Lawrence Erlbaum Associates, 1999.

Vince, J., *Virtual Reality Systems*: Addison-Wesley, 1995.

Visser, W., "Designing as construction of representations: a dynamic viewpoint in cognitive design research," *Human-Computer Interaction*, vol. 21 (1), pp. 103-152, 2006.

Vogel, D. and Balakrishnan, R., "Interactive public ambient displays: transitioning from implicit to explicit, public to personal, interaction with multiple users," *ACM symposium on user interface software and technology*, pp. 137-146, 2004.

Vries, B.d. and Achten, H.H., "DDDoolz: Designing with Modular Masses," *Design Studies*, vol. 23 (6), pp. 515-531, 2002.

Wielinga, B., Schreiber, A., & Breuker, J., "KADS: a Modeling Approach to Knowledge Engineering," *Knowledge Acquisition*, vol. 4 1-162, 1992.

Wolf, C.G., Carroll, J.M., Landauer, T.K., John, B.E., & Whiteside, J., "The Role of Laboratory Experiments in HCI: Help, Hindrance, or Ho-hum?," *SIGCHI Panel*, pp. 265-268, 1989.

Wolf, T.V., Rode, J.A., Sussman, J., & Kellogg, W.A., "Dispelling Design as the 'Black Art' of CHI," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 521-530, 2006.

Wonka, P., Wimmer, M., Sillioin, F., & Ribarsky, W., "Instant Architecture," *Transactions on Graphics (also in SIGGRAPH)* 669-677, 2003.

Yoon, J., Ryokai, K., Dyner, C., Alonso, J., & Ishii, H., "egaku: enhancing the sketching process," *SIGGRAPH*, pp. 2004.

Zeleznik, R.C., Herndon, K.P., & Hughes, J.F., "SKETCH: an interface for sketching 3D scenes," *SIGGRAPH'96*, pp. 163-70, 1996.

Zhai, S., Conversy, S., Beaudouin-Lafon, M., & Guiard, Y., "Human On-line Response to Target Expansion," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 177-184, 2003.

Zhai, S., Morimoto, C., & Ihde, S., "Manual and Gaze Input Cascaded (MAGIC) Pointing," *ACM Conference on Human Factors in Computing Systems (CHI)*, pp. 246-253, 1999.

Zhai, S. and Senders, J.W., "Investigating Coordination in Multidegree of Freedom Control I: Time-on-Target Analysis of 6 DOF Tracking," *41st Annual Meeting of Human Factors and Ergonomics Society*, pp. 1249-1254, 1997.

Appendix A Experiments IRB Approvals

Virginia Tech
VIRGINIA POLYTECHNIC INSTITUTE
AND STATE UNIVERSITY

Institutional Review Board

Dr. David M. Moore
IRB (Human Subjects)Chair
Assistant Vice President for Research Compliance
CVM Phase II- Duckpond Dr., Blacksburg, VA 24061-0442
Office: 540/231-4991; FAX: 540/231-6033
email: moored@vt.edu

DATE: October 19, 2004

MEMORANDUM

TO: Doug A. Bowman Computer Science 0106
Jian Chen CS 0106

FROM: David Moore DM
by TBM

SUBJECT: **IRB Exempt Approval:** "Design and Evaluation of Domain-Specific Objects manipulation Techniques for Massing Study in Virtual Environments" IRB # 04-512

I have reviewed your request to the IRB for exemption for the above referenced project. I concur that the research falls within the exempt status. Approval is granted effective as of October 15, 2004.

cc: File

Virginia Tech
VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY

Institutional Review Board
Dr. David M. Moore
IRB (Human Subjects) Chair
Assistant Vice President for Research Compliance
CVM Phase II- Duckpond Dr., Blacksburg, VA 24061-0442
Office: 540/231-4991; FAX: 540/231-6033
email: moored@vt.edu

DATE: May 12, 2005

MEMORANDUM

TO: Doug A. Bowman Computer Science 0106
Jian Chen CS 0106

FROM: David Moore

SUBJECT: **IRB Exempt Approval:** "Evaluation of Cloning Techniques in Virtual Environments" IRB # 05-343

I have reviewed your request to the IRB for exemption for the above referenced project. I concur that the research falls within the exempt status. Approval is granted effective as of May 12, 2005.

Virginia Tech has an approved Federal Wide Assurance (FWA00000572, exp. 7/20/07) on file with OHRP, and its IRB Registration Number is IRB00000667.

cc: File

A Land-Grant University - Putting Knowledge to Work
An Equal Opportunity/Affirmative Action Institution

Virginia Tech
VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY

Institutional Review Board
Dr. David M. Moore
IRB (Human Subjects) Chair
Assistant Vice President for Research Compliance
1880 Pratt Drive, Suite 2006(0497), Blacksburg, VA 24061
Office: 540/231-4991; FAX: 540/231-0959
email: moored@vt.edu

DATE: December 16, 2005

MEMORANDUM

TO: Doug A. Bowman Computer Science 0106
Jian Chen CS 0106

FROM: David Moore

SUBJECT: **IRB Expedited Approval:** "Evaluation of Interaction Techniques for Massing Study in Virtual Environments" IRB # 05-765

This memo is regarding the above-mentioned protocol. The proposed research is eligible for expedited review according to the specifications authorized by 45 CFR 46.110 and 21 CFR 56.110. As Chair of the Virginia Tech Institutional Review Board, I have granted approval to the study for a period of 12 months, effective December 16, 2005.

Virginia Tech has an approved Federal Wide Assurance (FWA00000572, exp. 7/20/07) on file with OHRP, and its IRB Registration Number is IRB00000667.

cc: File
OSP 0170

A Land-Grant University - Putting Knowledge to Work
An Equal Opportunity/Affirmative Action Institution

Appendix B Experiment I: Testbed Evaluation of Cloning Techniques in Virtual Environments

This experiment includes three studies. The first study evaluated several widgets user interfaces. According to users' comments, two further studies were conducted: one is to study the precision users can dragging sliders; the other study was to iterate on the first one. Redesign and evaluation of new user interfaces were conducted. In this dissertation, we only report the first study. Interested readers can refer to the technical report (Chen et al. 2004) for further information.

B.1 Testbed evaluation of Widgets Cloning User Interfaces

B.1.1 Informed consent

I. THE PURPOSE OF THIS RESEARCH/PROJECT

You are invited to participate in a study of interaction in virtual environments. This research studies the ways people act and work in a three-dimensional virtual world. This study involves experimentation for the purpose of evaluating and improving the user interface in virtual environment applications.

II. PROCEDURES

You will be asked to perform a set of tasks using a virtual environment system. These tasks consist of generating multiple copies of objects in a three-dimensional (3D) environment. You will wear a head-mounted display (HMD). Your role in these tests is that of an evaluator of the software. We are not evaluating you or your performance in any way; you are helping us to evaluate our system. All information that you help us attain will remain anonymous. The time you take to do each task and other aspects of your interaction with the system will be measured. You may be asked questions during and after the evaluation, in order to clarify our understanding of your evaluation.

You may also be asked to fill out a questionnaire relating to your background with such systems, and to take a short test of spatial ability.

The session will last about one hour. The tasks are not very tiring, but you are welcome to take rest breaks as needed. One scheduled rest break will be given to you about half-way through the experiment. You may also terminate your participation at any time, for any reason.

You will be given full instructions before every task. Because we are measuring a range of abilities, some tasks will be easy and others difficult. It is important that you understand the instructions before beginning each task. If anything is unclear, be sure to ask us questions.

III. RISKS

The proposed experiments are straightforward tests of performance using standard virtual environments displays, trackers, and input devices. Participation involves standing on an enclosed platform (while wearing the head-mounted display) and performing simple tasks. The physical components of these tasks are not stressful, and include head and body turning and pointing. All light and sound intensities are well within normal ranges. The only foreseeable physical risks are slight eye strain, dizziness, or mild nausea. There are no known mental risks.

If you experience any eyestrain, dizziness, or nausea during a session, then between tasks please remove the HMD and take a rest break. The experimenter will explain when you can take such rest breaks. If you are having trouble with any task, please tell us. If dizziness or nausea becomes uncomfortable, you will be allowed to leave with no penalty.

IV. BENEFITS OF THIS PROJECT

Your participation in this project will provide information that may be used to improve the design of virtual environments hardware and/or software. In this experiment, no guarantee of benefits has been made to encourage you to participate. You may receive a synopsis summarizing this research when completed. Please leave a self-addressed envelope with the experimenter and a copy of the results will be sent to you.

You are requested to refrain from discussing the evaluation with other people who might be in the candidate pool from which other participants might be drawn. For this experiment, please don't talk to other people before Feb 2004.

V. EXTENT OF ANONYMITY AND CONFIDENTIALITY

The results of this study will be kept strictly confidential. Your written consent is required for the researchers to release any data identified with you as an individual to anyone other than personnel working on the project. The information you provide will have your name removed and only a subject number will identify you during analyses and any written reports of the research.

The experiment may be video or audio taped. If it is taped, the tapes will be stored securely, viewed only by the experimenters (Doug A. Bowman & Jian Chen), and erased after 3 months. If the experimenters wish to use a portion of your video/audio tape for any other purpose, they will get your written permission before using it. Your signature on this form does not give them permission to show your video/audio tape to anyone else.

VI. COMPENSATION

Your participation is voluntary and unpaid.

VII. FREEDOM TO WITHDRAW

You are free to withdraw from this study at any time for any reason.

VIII. APPROVAL OF RESEARCH

This research has been approved, as required, by the Institutional Review Board for projects involving human subjects at Virginia Polytechnic Institute and State University, and by the Department of Computer Science.

IX. SUBJECT'S RESPONSIBILITIES AND PERMISSION

I voluntarily agree to participate in this study, and I know of no reason I cannot participate. I have read and understand the informed consent and conditions of this project. I have had all my questions answered. I hereby acknowledge the above and give my voluntary consent for participation in this project. If I participate, I may withdraw at any time without penalty. I agree to abide by the rules of this project

____ ______

Signature Date

____ ______________________

Name (please print) Contact: email or phone
(optional)

Should I have any questions about this research or its conduct, I may contact:

Investigator: Dr. Doug A. Bowman Phone (540) 231-2058
Professor, Computer Science Department (231-6931)
email: bowman@cs.vt.edu

Jian Chen Phone (540) 231-2641
Graduate Student, Computer Science Department (231-6931)
email: jichen8@vt.edu

Review Board: David M. Moore, Office of Research Compliance, CVM Phase II
(0442) 231-4991

B.1.2 Entrance Questionnaire (Demographic and Background Information)

Participant ID # __________
Date: ________________
Time: ________________

Thank you for participating in our study. Please let us know if you have any questions at any time. All information will be kept confidential and used for statistical purposes only.

Please answer the following questions:

1. Age: ______

2. Gender: (please circle) Male Female

3. What is your major? ______________________

4. Do you have normal vision? Yes (w/ Glasses Contacts) No

5. Are you right-handed or left-handed? Right Left

6. How do you rate your expert level in video games?

```
     1          2          3          4          5          6          7
     |----------|----------|----------|----------|----------|----------|
(novice)                                                          (expert)
```

7. Level of computer experience:

```
     1          2          3          4          5          6          7
     |----------|----------|----------|----------|----------|----------|
(novice)                                                          (expert)
```

8. Past experience with Virtual Environments: (please circle) Yes No

- If you circled yes, please describe the experience(s) briefly:

____
____
____

B.1.3 Interfaces ordered by participants for balanced design

1: Clone-from-parameter, 2: Clone-from-1D slider (cavalier projection), 3: Clone-from-1D slider 4: Clone-from-2D slider widgets, 5: Clone-from-3D slider widgets

Orders / Participants					
S1	4	5	2	1	3
S2	2	4	3	5	1
S3	3	1	5	4	2
S4	1	3	4	2	5
S5	5	2	1	3	4

B.1.4 Participants

Find two to three domain users from the architecture or construction department and two to three users from the center for HCI at Virginia Tech.

B.1.5 Tasks

1. (Matching task) Please use the selection box to select the objects marked “beam 1”, “beam 3” and “column 2” in the environment. Please create a new structure in the environment that has the same structure as the one shown on your tablet.

2. (Numeric task) Please select the column marked “column 2” from the structure. Then generate a new structure which should have 3 copies along x axes, 4 copies along y axes, and 3 copies along z axes. The distance between adjacent copies should be 10-unit along x axis, 30-unit along y axis and 5-unit along z axis. The direction of the cloned objects should be +x, -y, and –z.

B.1.6 During Questionnaire (Interview questions)

3. Can you describe what the interface means before I tell you about it? (Cognitive Affordance question)

4. Is it easy to set up the parameters? Why?

5. Is it easy to know those parameters that you already set up?

6. Do you feel that the interface distracts your attention?

7. (After finishing the gesture-based and cavalier project interface tasks) Do you like the gesture-based interface? Do you prefer the interface on which the visibility of the widgets is related to the gesture or do you prefer the interface on which all widgets are visible all the time? Why?

8. Do you like the additional 1D control added to the interface? Why?

9. Do you feel the tablet is cluttered with widgets and are you overwhelmed by their functionalities?

10. Are the scales marked on each axis helpful for you to complete your task? Did you look at the scales to move your slider?

11. Do you feel the parameters we use match your understanding of cloning? Is there anything missing?

12. What kinds of tasks would you expect to do using such user interfaces?

B.1.7 Exit Questionnaire

Participant ID # __________
Date: ________________
Time: ________________

Thank you for participating in our study. We would like you to answer the following questions about your experience today. They are very IMPORTANT questions. Please answer them carefully.

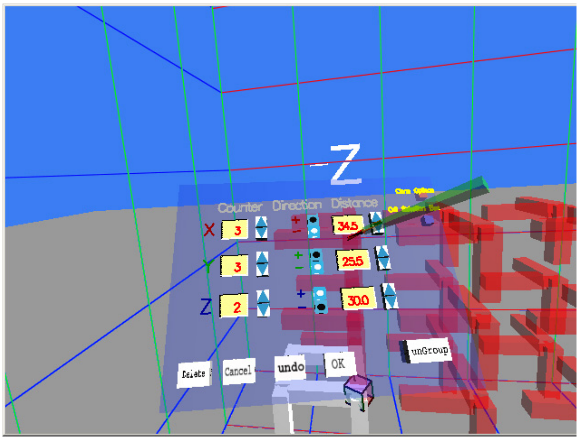
What you have experienced today is and application for cloning techniques in a virtual environment. The interfaces you have used in this experiment are named and pictured below. Please read them carefully because almost all questions are related to them. Please refer back or ask experimenter (Jian) questions at any time.

Techniques / Interfaces you have used:

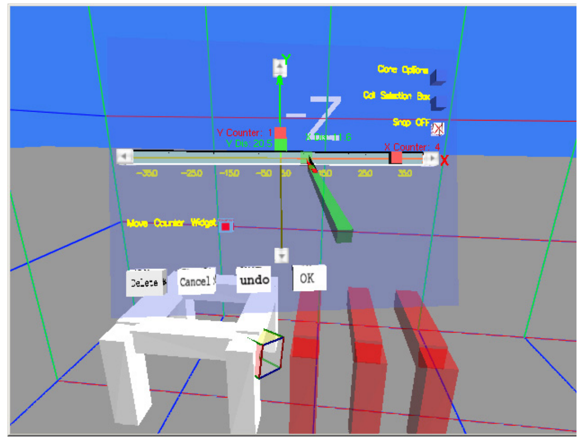
a. Clone-from-parameters
b. Clone-from-1D slider-widget
c. Clone-from- cavalier-projected-widget
d. Clone-from-2D-slider-widget
e. Clone-from-3D slider widget

Tasks you have performed:

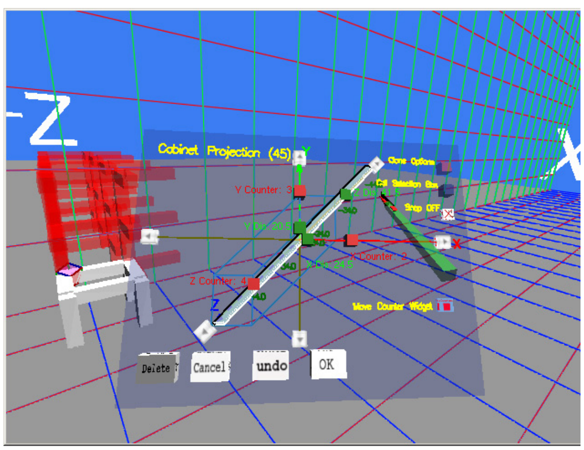
i. Task 1: matching task
ii. Task 2: numerical input task



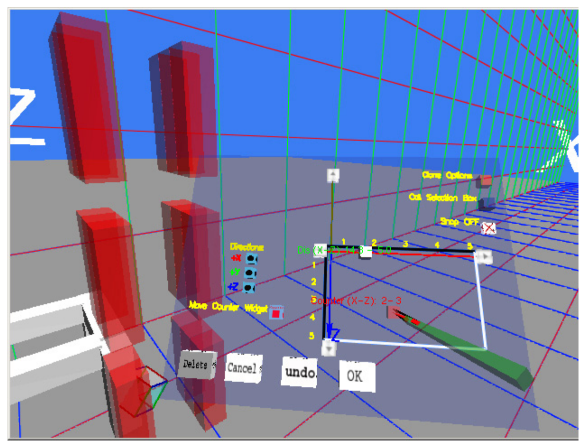
Clone-from-parameter



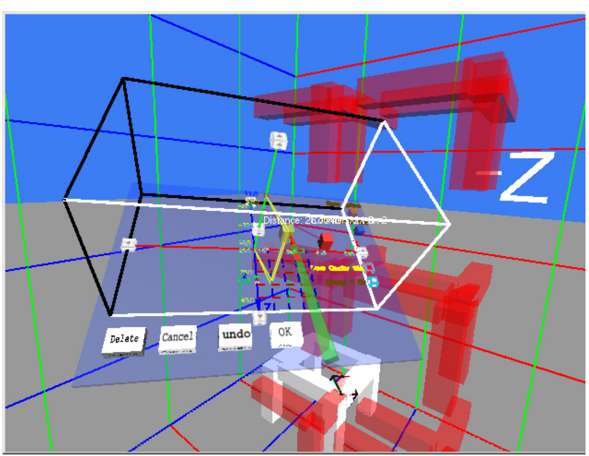
Clone-from-1D-slider widgets



Clone-from-cavalier projected widgets



Clone-from-2D-slider-widgets



Clone-from-3D-slider-widgets

Part one direction: Please circle or fill in the appropriate number corresponding to your opinion.

1. Please rate your preference of different interfaces using the scale provided below.

Like it very much 1	Like it 2	Somewhat like it 3	Neutral 4	Somewhat dislike it 5	Dislike it 6	Dislike it Very much 7

a. Parameter ( )
b. 1D-slider-widget ( )
c. Cavalier-projected-widget ( )
d. 2D-slider-widget ( )
e. 3D-slider-widget ( )

2. Please rate the overall level of difficulty of the tasks you completed in the Virtual Environment for each user interface.

Very Easy 1	Easy 2	Somewhat Easy 3	Neutral 4	Somewhat Difficult 5	Difficult 6	Very Difficult 7

f. Parameter

g. i. Task 1 _______ why?
h. ii. Task 2 _______ why?
i.
j. 1D-slider-widget

k. i. Task 1 _______ why?
l. ii. Task 2 _______ why?
m.
n. Cavalier-projected-widget

o. i. Task 1 _______ why?
p. ii. Task 2 _______ why?
q.
r. 2D-slider-widget

s. i. Task 1 _______ why?
t. ii. Task 2 _______ why?
u.
v. 3D-slider-widget

w. i. Task 1 _______ why?
x. ii. Task 2 _______ why?
y.

3. Please rate your level of learnabilities of using different interface to clone object using the scale provided below.

Very Easy 1	Easy 2	Somewhat Easy 3	Neutral 4	Somewhat Difficult 5	Difficult 6	Very Difficult 7

z. a. Parameter ( )
aa. b. 1D-slider-widget ( )
bb. Cavalier-projected-widget ( )
cc. 2D-slider-widget ( )
dd. 3D-slider-widget ( )
ee.
ff.

Part two directions: Please answer the following questions in detail. If you need more space, please write on the back of the sheet and include the question number

1. How well did you think the feedback provided improved your interaction with the environment? Did anything mislead you?

2. Any suggestions for improvement?

B.2 Law of dragging using slider widgets in a virtual environment

Participant ID # _______
Date: ________________
Time: ________________

B.2.1 Goal

The goal of this study is to get the proper resolution of the tick on widget user interfaces. We intended to measure quantitatively the width of the region that a user can accurately specify on a tablet-based slider UI given the current system setup.

The experiment is based on Fitts' law concept. Fitts' law is an information-theoretical view of human motor behavior. It express movement time (MT), the time to complete a movement task, in terms of the distance or amplitude of the move (A) and the widget of the region within which the move must terminate (W).

B.2.2 Entrance Questionnaire (Demographics and Background Info)

Thank you for participating in our study. Please let us know if you have any questions at any time. All information will be kept confidential and used for statistical purpose only.

Please answer the following questions:

Age: ___________

Gender (please circle): Male Female

Major: ___________

Do you have normal vision after correction? Yes No

Are you right-handed or left-handed? Right Left

How do you rate your level of video games experience?

Advanced Intermediate Novice

|-------------------|--------------------|

How do you rate your level of computer experience?

Advanced Intermediate Novice

|-------------------|--------------------|

Have you experienced virtual environments before? If yes, what type?

Desktop-based HMD-based CAVE-based

B.2.3 Participants

Two participants had architecture and engineering background and three users had HCI background. All were graduate students at Virginia Tech.

B.2.4 Design

The two independent variables were the region width (W) and the direction of the sliders, either horizontal (for X) or vertical direction (for Y). The dependent variable was the task completion time. We fixed the amplitude of the movement (A) at 150mm. The testing conditions for W were W = 30, 15, 7.5, 3.75, 2.5, and 1.875mm. A balanced within-subjects design was used. We used six participants in our experiment. They were randomly assigned to different conditions using a Latin square pattern to a unique order.

B.2.5 Tasks

Six participants performed numeric input task by dragging the slider from the origin to the desired position. They performed six tasks given the six different Ws. The task description was displayed on the screen during the experiment for the user's reference.

Two females and four males participated this study.

B.2.6 Result

The overall task completion time of the six participants under six conditions follow.

ID / w	1	2	3	4	5	6
10	2.5	2.62	1.63	1.59	2.36	2.67
20	3.43	2.65	2.2	2.25	2.42	1.86
40	4.71	2.97	2.33	1.94	1.78	2.13
80	2.37	7.43	7.84	4.61	2.98	1.75
120	6.11	13.52	4.06	13.73	3.86	5.79
160	7.66	10.03	5.15	16.52	4.75	5.45

We preformed a two-factor Analysis of Variance (ANOVA) on the width of the region (W) and the direction of the motion (X is horizontal and Y is vertical). The results were: (1) the widths of the regions were significant ($F=10.16$, $p<0.0001$); (2) the directions were not significant ($F=0.71$, $p=0.4$); (3) the two-way interaction was not significant. By performing the least significant difference (LSD) test on W, we found three groups, 30 and 15 (group 1); 7.5 and 3.75 (group 2); and 1.875 (group 3). We did not have enough power to decide which group W=2.5 belonged to, either group 2 or group 3.

We chose 15mm as the most appropriate region width for our future interface design. This width provides 10 distinct positions on the sliders for all three directions because of the physical size limitation of the tablet (480mm x 350mm) and the space taken by other widgets.

Appendix C Experiment II: Testbed evaluation of Cloning User interfaces

C.1 Effects of Cloning Tasks

We compared five cloning techniques. They are keypad, PORT, space-filling metaphor, copy-by-example, and dynamic slider for cloning.

C.1.1 Questionnaires

C.1.1.1 Entrance Questionnaire

(The same as B.1.2)

C.1.1.2 Exit Questionnaire

Thank you for participating in our study. We would like you to answer the following questions about your experience today. They are very IMPORTANT questions. Please answer them carefully.

What you have experienced today is and application for cloning techniques in a virtual environment. The interfaces you have used in this experiment are named and pictured below. Please read them carefully because almost all questions are related to them. Please refer back or ask experimenter (Jian) questions at any time.

Techniques / Interfaces you have used:

a. Keypad
b. PORT
c. Space-filling metaphor
d. Copy-by-example
e. Dynamic slider for cloning

Part one direction: Please circle or fill in the appropriate number corresponding to your opinion.

1. Please rate your preference of different interfaces using the scale provided below.

Like it very much 1	Like it 2	Somewhat like it 3	Neutral 4	Somewhat dislike it 5	Dislike it 6	Dislike it Very much 7

gg. Keypad ( )
hh. PORT ( )
ii. Space-filling metaphor ( )
jj. Copy-by-example ( )
kk. Dynamic slider for cloning ( )

2. Please rate the perceived usefulness of the user interfaces. (Perceived usefulness is defined as the degree to which you believe that using this interface would enhance your job performance).

Very Easy 1	Easy 2	Somewhat Easy 3	Neutral 4	Somewhat Difficult 5	Difficult 6	Very Difficult 7

ll.

a. Keypad ( )
b. PORT ( )
c. Space-filling metaphor ( )
d. Copy-by-example ( )
e. Dynamic slider for cloning ( )

mm.

3. Please rate the perceived ease of use of the user interfaces.

Very Easy 1	Easy 2	Somewhat Easy 3	Neutral 4	Somewhat Difficult 5	Difficult 6	Very Difficult 7

nn.

a. Keypad ( )
b. PORT ( )
c. Space-filling metaphor ( )
d. Copy-by-example ( )
e. Dynamic slider for cloning ( )

oo.
pp.

C.1.2 Tasks

Task 1 (building task) required participants to add two pieces one by one to an existing structure, then to raise the building up to four stories.

Task 2 (numeric input task) required participants to model a large structure. An example task was “Select that one column and two beams in the environment. Construct a two story building with 5 by 5 bays on the horizontal plane.”

Task 3 (picture matching task) required the participant to build a structure to match an existing design displayed as a two-dimensional image on the tablet. This task corresponds to the situation when the designer has finished a draft on a paper and brought it to a VE to construct the structure.

C.1.3 Results (HOMER-C vs. PORT)

C.1.3.1 Participants’ demographics information

	Age	Game experience	Computer experience	Hours of computer use
S1	22	5	6	5
S2	20	6	4	3
S3	20	5	4	5
S4	21	3	5	3
S5	20	4	5	3
S6	27	1	6	5

S7	23	3	5	2

C.1.3.2 ETS standard psychology paper test results

	Number comparison test			Maze tracking test			Pattern matching test		
	T1	T2	*Sum*	T1	T2	*Sum*	T1	T2	*Sum*
S1	22	26	*48*	12	14	*26*	18	12	*30*
S2	18	22	*40*	15	15	*30*	9	9	*18*
S3	40	40	*80*	15	18	*33*	7	7	*14*
S4	18	20	*38*	20	19	*39*	18	20	*38*
S5	26	34	*60*	19	19	*38*	9	9	*18*
S6	20	22	*42*	23	24	*47*	15	15	*30*
S7	24	30	*54*	19	20	*39*	17	15	*32*

C.1.3.3 Preference

The scale is from 1 to 7, the higher the better.

	Building task		Numerical input task		Matching task	
	PORT	HOMER	PORT	HOMER	PORT	HOMER
S1	7	1	7	1	7	1
S2	5	5	5	3	5	5
S3	3	7	7	1	6	7
S4	3	2	6	2	7	2
S5	6	5	7	2	5	4
S6	6	5	6	4	6	3
S7	4	6	7	3	6	5

C.1.3.4 Perceived usefulness (enhance job performance)

	Building task		Numerical input task		Matching task	
	PORT	HOMER	PORT	HOMER	PORT	HOMER
S1	7	3	7	3	7	1
S2	5	5	5	3	5	5
S3	5	5	3	1	6	7
S4	4	4	3	4	4	2
S5	6	6	6	4	7	4
S6	5	3	6	5	6	3
S7	5	6	4	2	5	5

C.1.3.5 Perceived ease of use (free of physical and mental efforts)

	Building task		Numerical input task		Matching task	
	PORT	HOMER	PORT	HOMER	PORT	HOMER
S1	6	4	6	4	6	4
S2	3	5	4	5	5	3
S3	7	7	7	1	7	5
S4	7	3	6	3	7	3
S5	4	7	6	3	6	2

S6	5	4	6	5	7	3
S7	5	7	4	2	6	5

C.2 Effects of Cloning Techniques

C.2.1.1 Performance (all data)

Keypad: keypad for cloning; PORT: port for cloning; SM: space-filling metaphor for cloning; CE: copy-by-example for cloning; Slider: dynamic slider for cloning.

The numbers in the table are participants' performance. The unit is second.

	Building task					Numerical input task					Matching task				
	Keypad	PORT	SM	CE	Slider	Keypad	PORT	SM	CE	Slider	Keypad	PORT	SM	CE	Slider
S1	25.2	68	121			54	41	80				70	122		
S2	22	87	66			47	48	70				171	133		
S3	28	57	62			40	30	34				80	105		
S4	38	88	79			53	31	35				70	87		
S5	75	109.8	123			96	31	100				123	214		
S6	33	92	61			104	106	64				123	149		
S7	75	77	100			106	56	67				148	233		
S8	56	65	89			100	42	38				63	93		
S9			373.6	75	380			75.7	331	68			102	130	130
S10			88	80	152			50	61	43			100	105	146
S11			88	90	184			50	56	40			101	296	139
S12			122	76	149			42	121	44			80	164	103
S13			63	50	81			30	54	31			52	78	71
S14			94	110	210			51	132	66			111	150	149
S15			122	69	189			30	65	40			106	87	95
S16			373.6	75	380			75.7	331	68			102	130	130

C.2.1.2 Performance (mean task completion time)

	Cloning	Selection	Overall	Cloning	Selection	Overall	Cloning	Selection	Overall
	Building task			Numerical input task			Matching task		
PORT	29.8	10.9	80.2	24	6.05	48.2	31.4	20.1	106
Space-filling	44.7	10.8	90.4	39.6	4.06	50.3	36.8	19.7	120.58
D-slider	76.2	9.7	113.05	24.3	3.1	47.0	69.9	20.7	113.4
copybyExample	10.6	13.8	72.4	27.7	11.14	79.1	13.2	17.7	106.14
keypad	56.3	14.5	106	42.9	6.78	74.6	x	x	x

Appendix D Experiment III: Comparison of Virtual-SAP 1.0 and 2.0

D.1 Questionnaires

D.1.1 Entrance Questionnaire

The questionnaire was web-based at http://www.survey.vt.edu. Participants filled out this form online.

Name:
Email address:

Gender: Male Female

Major / level:

for how many years?

How do you rate your level of expertise in structure engineering?
1. novice
2.
3.
4. neutral
5.
6.
7. expert

How many hours do you use computer daily?
0-1 1-2 3-4 above 5

What computer aided design tool have you used?
AutoCAD
SAP
SketchUp
other:

Please rate your experience with video games?
1 (novice) 2 3 4 5 6 7 (expert)

Describe your past experience with virtual environments (if you never experienced VEs before, just answer No.)

D.1.2 Exit questionnaire

The questionnaire was online on http://www.survey.vt.edu. Participants filled out the form online.

Please answer the following question based on today's experiment.

Name:
Date (mm/dd/yyyy) and time:

1. Do you agree that the user interface matches the work domain constraints (e.g., shape, size, language used and objects' behavior
1 strongly disagree
2 disagree
3 somewhat disagree
4 neutral
5 somewhat agree
6 agree
7 strongly agree
Comments:

2. Do you agree that using the interfaces is mentally challenging?
1 strongly disagree
2 disagree
3 somewhat disagree
4 neutral
5 somewhat agree
6 agree
7 strongly agree
Why:

3. Do you agree that information presented in the interface is overwhelming?
1. strongly disagree
2. disagree
3. somewhat disagree
4. neutral
5. somewhat agree

6. agree
7. strongly agree
Why? What information were you looking for when you create or move or select object(s)?

4. Selecting objects in the virtual world is:
1. very difficult
2. difficult
3. somewhat difficult
4. neutral
5. somewhat easy
6. easy
7. very easy

5. Selecting widgets on the tablet is:
1. very difficult
2. difficult
3. somewhat difficult
4. neutral
5. somewhat easy
6. easy
7. very easy

6. Copying and pasting object(s) is:
1. very difficult
2. difficult
3. somewhat difficult
4. neutral
5. somewhat easy
6. easy
7. very easy

7. Moving an object is:
1. very difficult
2. difficult
3. somewhat difficult
4. neutral
5. somewhat easy
6. easy
7. very easy

8. Moving a group of objects:
1. very difficult
2. difficult
3. somewhat difficult

4. neutral
5. somewhat easy
6. easy
7. very easy

9. Selecting object(s) is:
1. very difficult
2. difficult
3. somewhat difficult
4. neutral
5. somewhat easy
6. easy
7. very easy

10. Is it easy to build a structure in your mind?
1. very difficult
2. difficult
3. somewhat difficult
4. neutral
5. somewhat easy
6. easy
7. very easy

11. Would you agree that the interface is intuitive?
1. strongly disagree
2. disagree
3. somewhat disagree
4. neutral
5. somewhat agree
6. agree
7. strongly agree

12. Would you agree that the interface is something that you want to use for design?
1. strongly disagree
2. disagree
3. somewhat disagree
4. neutral
5. somewhat agree
6. agree
7. strongly agree
Why?

13. What was the most useful technique that helped you finish your task?

14. What is the worst technique designed that hindered your modeling process?

15. Any functions are missing from the user interface? How can we improve the interface to make the design better?

D.2 User behaviors classification

Behavior		
Task-oriented		Asks for opinion
		Ask for suggestions
		Ask for information
		Give suggestions
		Give opinions
		Give information
Social-emotional Behavior		
	Positive	Shows solidarity
		Dramatization
		Agrees
	Negative	Disagrees
		Shows tension
		Shows antagonisms

D.3 Overall task completion time

Participant ID	Task ID	Group ID	Performance (s)
s1	t1	group1	600
s2	t1	group1	346
s3	t1	group1	453
s4	t1	group1	327
s5	t1	group1	449
s6	t1	group2	93
s7	t1	group2	42
s8	t1	group2	134
s9	t1	group2	90
s10	t1	group2	140
s1	t2	group1	124
s2	t2	group1	659
s3	t2	group1	330
s4	t2	group1	470
s5	t2	group1	770
s6	t2	group2	313
s7	t2	group2	121
s8	t2	group2	146
s9	t2	group2	278
s10	t2	group2	154
s1	t3	group1	740
s2	t3	group1	800
s3	t3	group1	952
s4	t3	group1	1140

s5	t3	group1	1013
s6	t3	group2	926
s7	t3	group2	377
s8	t3	group2	167
s9	t3	group2	157
s10	t3	group2	170

Appendix E Experiment IV: Evaluation of Massing Study User Interface

All form and questionnaires were web-based on http://www.survey.vt.edu. Participants filled in the forms online before or after the experiment.

E.1 Informed consent

Informed Consent for Participant of Investigative Project

Virginia Polytechnic Institute and State University

Title of Project:
Evaluation of 3D User Interfaces for Massing Study in Virtual Environments

Principal Investigator:
Dr. Doug A. Bowman; Jian Chen

I. THE PURPOSE OF THIS RESEARCH/PROJECT
You are invited to participate in a study of interaction in virtual environments. This research studies the ways people act and work in a three-dimensional virtual world. This study involves experimentation for the purpose of evaluating and improving the user interface in virtual environment applications.

II. PROCEDURES
You will be asked to perform a set of tasks using a virtual environment system. These tasks consist of modeling in 3D environment. You will be using 3D trackers and view the image from a large-screen display. Your role in these tests is that of evaluator of the software. We are not evaluating you or your performance in any way; you are helping us to evaluate our system. All information that you help us attain will remain anonymous. The time you take to do each task and other aspects of your interaction with the system will be measured. You may be asked questions during and after the evaluation, in order to clarify our understanding of your evaluation.

You may also be asked to fill out a questionnaire relating to your background with such systems, and to take a short test of spatial ability.

The session will last about one hour. The tasks are not very tiring, but you are welcome to take rest breaks as needed. One scheduled rest break will be given to you about half-way through the experiment. You may also terminate your participation at any time, for any reason.

You will be given full instructions before every task. Because we are measuring a range of abilities, some tasks will be easy and others difficult. It is important that you understand the instructions before beginning each task. If anything is unclear, be sure to ask us questions.

III. RISKS

The proposed experiments are straightforward tests of performance using a large screen display and input devices. Participation will be sitting all the time and performing simple tasks. The physical components of these tasks are not stressful, and include normal gestures we do everyday. All light and sound intensities are well within normal ranges. There are no known mental and physical risks.

If you experience any eyestrain, dizziness, or nausea during a session, then between tasks please take a rest break. If you are having trouble with any task, please tell us. If dizziness or nausea becomes uncomfortable, you will be allowed to leave with no penalty.

IV. BENEFITS OF THIS PROJECT

Your participation in this project will provide information that may be used to improve the design of virtual environments hardware and/or software. You will be compensate $10 / hour for your time. You may receive a synopsis summarizing this research when completed. Please leave a self-addressed envelope with the experimenter and a copy of the results will be sent to you.

You are requested to refrain from discussing the evaluation with other people who might be in the candidate pool from which other participants might be drawn. For this experiment, please don't talk to other people before Oct 6th 2006.

V. EXTENT OF ANONYMITY AND CONFIDENTIALITY

The results of this study will be kept strictly confidential. Your written consent is required for the researchers to release any data identified with you as an individual to anyone other than personnel working on the project. The information you provide will have your name removed and only a subject number will identify you during analyses and any written reports of the research.

The experiment may be video or audio taped. If it is taped, the tapes will be stored securely, viewed only by the experimenters (Doug A. Bowman, Jian Chen), and erased after 3 months. If the experimenters wish to use a portion of your video/audio tape for any other purpose, they will get your written permission before using it. Your signature on this form does not give them permission to show your video/audio tape to anyone else.

VI. COMPENSATION

Your participation is paid $10 / hour.

VII. FREEDOM TO WITHDRAW
You are free to withdraw from this study at any time for any reason.

VIII. APPROVAL OF RESEARCH
This research has been approved, as required, by the Institutional Review Board for projects involving human subjects at Virginia Polytechnic Institute and State University, and by the Department of Computer Science.

IX. SUBJECT'S RESPONSIBILITIES AND PERMISSION
I voluntarily agree to participate in this study, and I know of no reason I cannot participate. I have read and understand the informed consent and conditions of this project. I have had all my questions answered. I hereby acknowledge the above and give my voluntary consent for participation in this project. If I participate, I may withdraw at any time without penalty. I agree to abide by the rules of this project.

Would you agree to participate this study?
Yes
No

Name:

Date:

Email address:

E.2 Questionnaires

E.2.1 Entrance questionnaire

Name:
Email address:

Gender: Male Female
Age:

Major / level:
for how many years?

What modelling tool have you used?

AutoCAD
3D studio max
SketchUp
other:

Please rate your experience with video games?
1 (novice)   2   3   4   5   6   7 (expert)

Describe your past experience with virtual environments (if you never experienced VEs before, please just answer No.)

E.2.2 Exit questionnaire

1. Please rate the perceived usefulness of the user interface. (Perceived usefulness is defined as the degree to which you believe that using this interface would enhance your job performance).

1. Very helpful
2. Helpful
3. Somewhat helpful
4. Neutral
5. Not much helpful
6. Not helpful
7. Not helpful at all

Comments:

2. Do you think that using the interfaces is mentally challenging?

1. Strongly disagree
2. Disagree
3. Somewhat disagree
4. Neutral
5. Somewhat agree
6. Agree
7. Strongly agree

Why:

3. Placing an object to a desired location was:
1. very difficult
2. difficult
3. somewhat difficult
4. neutral

5. somewhat easy
6. easy
7. very easy

4. Getting an copy of an object was:

1. very difficult
2. difficult
3. somewhat difficult
4. neutral
5. somewhat easy
6. easy
7. very easy

5. Would you agree that physics was helpful for modeling?
1. Strongly disagree
2. Disagree
3. Somewhat disagree
4. Neutral
5. Somewhat agree
6. Agree
7. Strongly agree

6. What was most useful part of the user interface?

7. What was the worst part of the user interface that hindered your design work ?

8. Additional comments?

Vita

Jian Chen was born in China on October 11, 1973. She received her Bachelor of Science degree in Precision Instruments and Methodology from Changchun University of Science and Technology in 1993. She also earned a Master of Engineering degree in Mechanical Engineering from Tianjin University and Tsinghua University in 1999. She has received numerous awards and honors, including the National Challenge Cup (Tiaozhan Cup) for Excellence in Design in 1999 and the best paper award from HFES (Human Factor and Erognomic Society) in 2003, membership in Upsilon Pi Epsilon, etc. Her work has been presented at conferences including IEEE Virtual Reality, VRST, HFES, Graphics Interface, I3D, 3DUI workshop, and in journal including International Journal of Virtual Reality.