

Advanced Topics in 3D User Interface Design

SIGGRAPH 2001
August 14, 2001
Course Notes



Doug A. Bowman
Virginia Tech

Joseph J. LaViola Jr.
Brown University

Mark R. Mine
Walt Disney Imagineering

Ivan Poupyrev
Sony Computer Science Labs

Course Overview

Three-dimensional (3D) interaction is an exciting field of research that promises to allow users to perform tasks freely in three dimensions rather than being limited by the 2D desktop metaphor of conventional graphical interfaces. Applications of immersive and desktop virtual environments (VEs), augmented reality (AR), and ubiquitous computing all require efficient and usable 3D interfaces. However, spatial interaction is not well-understood and presents significant new challenges that are not addressed satisfactorily by traditional 2D human-computer interaction (HCI) research. Some 2D techniques have proved useful when implemented in 3D, but these are not sufficient.

This course builds on a course offered at SIGGRAPH 2000 titled 3D User Interface Design: Fundamental Techniques, Theory, and Practice. That course focused on the basic input and output devices and interaction techniques for 3D applications. Here, we will present a more detailed discussion of implementation issues and strategies, including the mathematical basis for various techniques and software tools that can be used for implementation. We will also discuss several strategies for 3D interface design, and methods for integrating diverse interaction techniques into applications. We will also present methods for usability evaluation of 3D interfaces, and finally will include a special section on 3D interaction in specialized interface styles such as augmented reality and ubiquitous computing.

Speaker Biographies



Doug A. Bowman is an Assistant Professor of Computer Science at the Virginia Polytechnic Institute and State University (Virginia Tech). Doug's research interests are in the intersection between human-computer interaction and computer graphics. Specifically, he is studying interaction and user interfaces in immersive virtual environments, and focusing on the design, evaluation, and application of 3D interaction techniques for complex VEs. His work has appeared in journals such as *Presence* and *IEEE Computer Graphics & Applications*, and he has presented research at conferences including ACM SIGGRAPH, the ACM Symposium on Interactive 3D Graphics, and IEEE Virtual Reality. Doug is the co-founder of the Virginia Tech Center for Virtual Environments and Visualization (CVEV). He is a member of ACM and the IEEE Computer Society. Doug received his M.S. and Ph.D. degrees in Computer Science from the Georgia Institute of Technology.



Joseph J. LaViola Jr. is currently a Ph.D candidate in the Computer Science department at Brown University. He works under the direction of Andries van Dam and Robert Zeleznik in the Computer Graphics Group. His primary research interests include interaction and visualization of large scale datasets, multimodal interaction in virtual environments and user interface evaluation. He holds masters degrees in Computer Science and Applied Mathematics from Brown University. He has received a number of awards which include Florida Atlantic University's Aaron Finerman Award, the Microsoft Senior Achievement Award, and is currently supported by the van Dam Fellowship. He is a member of Phi Eta Sigma, Phi Kappa Phi, and associate member of Sigma Xi. In addition to his academic pursuits, he also runs J.J.L. Interface Consultants, a consulting business specializing in interfaces for VR applications.



Ivan Poupyrev is a researcher at the Sony Computer Science Laboratories in Tokyo. Prior to joining Sony he spent two years at the Media Integration and Communication Research Laboratories of ATR International in Kyoto during and after work on his doctorate in Computer Science at Hiroshima University. He spent two years at the Human Interface Technology Laboratory at the University of Washington as a Visiting Scientist designing and investigating 3D user interfaces. His current research interests are in designing and investigating advanced interfaces between humans and machines, including 3D interfaces, augmented reality interfaces, ubiquitous and wearable interfaces. Results of his work have been presented at conferences such as UIST, CHI, SIGGRAPH, EUROGRAPHICS, VRAIS and others. He received an M.S. in Applied Mathematics and Computer Science from Moscow Airspace University, Soviet Union.



Mark R. Mine is a senior member of the technical staff at Walt Disney Imagineering Research & Development Inc. Currently he is working for WDI's VR Studio, developing tools for the creation and manipulation of virtual spaces. He has a B.S. in Aerospace Engineering from the University of Michigan, an M.S. in Electrical Engineering from the University of Southern California, and a Ph.D. in Computer Science from the University of North Carolina. His Ph.D. dissertation, "Exploiting Proprioception in Virtual-Environment Interaction" (written under the direction of Dr. Frederick P. Brooks Jr.) explores the benefits and limitations of working in a virtual world.

Schedule

Morning:

- 8:30 Welcome and Introduction – Bowman
- 8:40 Review of 3D Technology – LaViola
- 9:15 Basic 3D Interaction Techniques – Bowman
- 10:15 Break
- 10:30 Advanced 3D Interaction Techniques – LaViola
- 11:00 Interaction Metaphors and Strategies – Poupyrev
- 11:30 Adapting 2D Interfaces to 3D Applications – LaViola
- 12:00 Lunch

Afternoon:

- 1:30 3D Interaction in Non-Immersive Environments – Poupyrev
- 2:30 Evaluation of 3D Interfaces – Bowman
- 3:15 Break
- 3:30 Industrial Perspective on 3D Interfaces – Mine
- 4:30 Panel Discussion – All

Table of Contents

PART I: Lecture Slides and Notes

1. **Welcome and Introduction** -- Doug Bowman
2. **A Review of Input and Output Devices for 3D Interaction** – Joseph LaViola
3. **Basic 3D Interaction Techniques** – Doug Bowman
4. **Non-Isomorphic Interaction in 3D User Interfaces** – Joseph LaViola
5. **3D Interaction Strategies and Metaphors** – Ivan Poupyrev
6. **Bringing 2D Interfaces into 3D Worlds** – Joseph LaViola
7. **Beyond VR: 3D Interfaces in Non-immersive Environments** – Ivan Poupyrev
8. **Evaluation of 3D Interfaces** – Doug Bowman
9. **Virtual Disney Worlds** – Mark Mine
10. **Color Plates**

PART II: Papers

Input and Output Devices

“HMDs, Caves, and Chameleon: A Human-Centric Analysis of Interaction in Virtual Space”, Bill Buxton and George W. Fitzmaurice

“Flex and Pinch: A Case Study of Whole Hand Input Design for Virtual Environment Interaction”, Joseph LaViola and Robert Zeleznik

“Investigating Coordination in Multidegree of Freedom Control I: Time-on-Target Analysis of 6 DOF Tracking”, Shumin Zhai and John Senders

“Investigating Coordination in Multidegree of Freedom Control II: Correlation Analysis in 6 DOF Tracking”, Shumin Zhai and John Senders

“Quantifying Coordination in Multiple DOF Movement and its Application to Evaluating 6 DOF Input Devices”, Shumin Zhai and Paul Milgram

Advanced 3D Interaction Techniques

“Hands-Free Multi-Scale Navigation in Virtual Environments”, Joseph LaViola, Daniel Feliz, Daniel Keefe, and Robert Zeleznik

“Non-Isomorphic 3D Rotational Techniques”, Ivan Poupyrev, Suzanne Weghorst, and Sidney Fels

2D Interaction in a 3D World

“ErgoDesk: A Framework for Two- and Three-Dimensional Interaction at the ActiveDesk”, Andrew S. Forsberg, Joseph J. LaViola Jr., and Robert C. Zeleznik

“Design and Evaluation of Menu Systems for Immersive Virtual Environments”, Doug Bowman and Chad Wingrave

“A Multi-Layered Architecture for Sketch-Based Interaction within Virtual Environments”, Oliver Bimber, L. Miguel Encarnacao, and Andre Stork

3D Interfaces for Non-Immersive Environments

“Mixing Realities in Shared Space: An Augmented Reality Interface for Collaborative Computing”, Mark Billinghurst, Ivan Poupyrev, Hirokazu Kato, and Richard May

“Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments”, Jun Rekimoto and Masanori Saitoh

“InfoStick: An Interaction Device for Inter-Appliance Computing”, Naohiko Kohtake, Jun Rekimoto, and Yuichiro Anzai

“Tiles: A Mixed Reality Authoring Interface”, Ivan Poupyrev, Desney Tan, Mark Billinghurst, Hirokazu Kato, Holger Regenbrecht, and Nobuji Tetsutani

“Virtual Object Manipulation on a Table-Top AR Environment”, H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana

Evaluation of 3D Interfaces

“Usability Evaluation in Virtual Environments: A Comparison and Integration of Methods”, Doug Bowman, Joseph Gabbard, and Deborah Hix

Virtual Disney Worlds

“Designing Interactive Theme Park Rides: Lessons Learned Creating Disney’s Pirates of the Caribbean – Battle for the Buccaneer Gold”, Jesse Schell and Joe Shochet

“Disney’s Aladdin: First Steps Toward Storytelling in Virtual Reality”, Randy Pausch, Jon Snoddy, Robert Taylor, Scott Watson, and Eric Haseltine

PART III: Annotated Bibliography

“20th Century 3DUI Bib: Annotated Bibliography of 3d User Interfaces of the 20th Century”, compiled by Ernst Kruijff and Ivan Pouprjev

Part I: Lecture Slides and Notes



Welcome to SIGGRAPH 2001, and to the course on “Advanced Topics in 3D User Interface Design”.

Welcome & Introduction

- **This course covers advanced topics related to the design and evaluation of 3D UIs**
- **Focus is on**
 - software techniques
 - implementation of usable interfaces
- **Focus is not on**
 - technology (but we will review devices)
 - 3D applications with 2D interfaces

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Three-dimensional (3D) interaction is an exciting field of research that promises to allow users to perform tasks freely in three dimensions rather than being limited by the 2D desktop metaphor of conventional graphical interfaces. Applications of immersive and desktop virtual environments (VEs), augmented reality (AR), and ubiquitous computing all require efficient and usable 3D interfaces. However, spatial interaction is not well-understood and presents significant new challenges that are not addressed satisfactorily by traditional 2D human-computer interaction (HCI) research. Some 2D techniques have proved useful when implemented in 3D, but these are not sufficient. Therefore, we offer this course highlighting the current state of the art.

The course is advanced, so it assumes that the attendees have some background in developing 3D systems, in basic human-computer interaction and user interfaces, and a basic knowledge of common 3D interaction techniques. We will focus here on interaction techniques, not on technology/hardware (however, there will be a review of some important and novel 3D input/output devices, with a view towards their use in interfaces). We are talking about truly 3D interfaces, not 2D interfaces to 3D applications (e.g. 3D modeling packages on the desktop).

Last year's course

- **Last year: "3D UI Design: Fundamental Techniques, Theory, and Practice"**
 - Survey of technology and techniques
 - Heavy emphasis on immersive VEs
- **This year**
 - In-depth look at common techniques and their implementations
 - Interfaces for AR, desktop 3D
 - More on evaluation

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

This course builds on a course offered at SIGGRAPH 2000 titled 3D User Interface Design: Fundamental Techniques, Theory, and Practice. That course focused on the basic input and output devices and interaction techniques for 3D applications. Here, we will present a more detailed discussion of implementation issues and strategies, including the mathematical basis for various techniques and software tools that can be used for implementation. We will also discuss several strategies for 3D interface design, and methods for integrating diverse interaction techniques into applications. We will also present methods for usability evaluation of 3D interfaces, and finally will include a special section on 3D interaction in specialized interface styles such as augmented reality and desktop 3D interaction.

Why 3D Interaction?

- **3D/VE applications should be useful**
 - immersion
 - natural skills
 - immediacy of visualization
- **But, applications in common use have low complexity of interaction**
- **More complex applications have serious usability problems**
- **Technology alone is not the solution!**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Three dimensions and virtual environments intuitively make sense for a wide range of applications, because of the characteristics of the tasks and their match with the characteristics of these environments. Immersion is the feeling of “being there” (replacing the physical environment with the virtual one), which makes sense for applications such as training and simulation. If a user is immersed *and* can interact using natural skills, then the application can take advantage of the fact that the user already has a great deal of knowledge about the world. The immediacy characteristic refers to the fact that there is a short “distance” between a user’s action and the system’s feedback that shows the result of that action. This can allow users to build up complex mental models of how a simulation works, for example.

Most applications in common use (e.g. walkthroughs, psychiatric treatment, entertainment, and training) contain user interaction which is not very complex. Other types of applications (e.g. immersive design, education, complex scientific visualizations) are for the most part still stuck in the research lab, often because they have usability problems that limit their usefulness.

Better technology is not the only answer - 30 years of VE technology research have not ensured that today’s VEs are usable - we must also focus on the design of interaction for VEs.

Therefore, we feel that 3D interaction is a vital topic for all 3D/VE developers, designers, and evaluators to understand.

Definitions

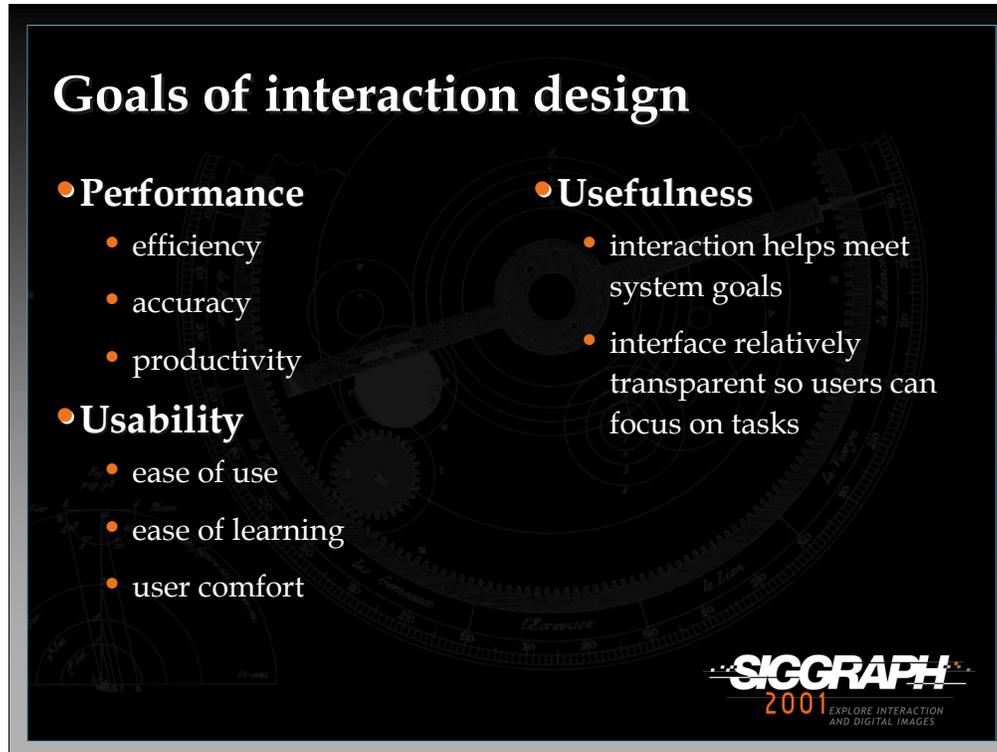
- **Interface**
 - the hardware/software through which a user communicates with a computer system and vice-versa
- **Input device**
 - the hardware component of user-computer communication
- **Interaction technique (IT)**
 - a method by which the user accomplishes some system task via the interface

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The user interface is simply a communications medium. The user communicates to the system via input of various types. The system presents information to the user via displays/output. 3D interfaces have the potential to greatly increase the bandwidth of the communications from the user to the system and from the system to the user.

An input device is simply some piece of hardware that is used to communicate with the system (e.g. mouse, keyboard, cyberglove, stylus, touch screen, etc.). We will discuss 3D input devices in the first lecture.

An interaction technique is part of the user interface (UI). It is a method that allows the user to perform some task in the system, and it includes both hardware (input device) and software components.



We will try to keep in mind as we discuss ways to accomplish 3D interaction tasks that we want to design for performance, usability, and usefulness.

Performance relates to quantitative measures indicating how well the task is being done by the user and the system in cooperation. This includes standard metrics like efficiency and accuracy.

Usability refers to the ease of communicating the user's intentions to the system, and the qualitative experience of the user.

Usefulness implies that the system is actually helping the user perform work or meet his/her goals, without being hindered by the interface.

All three of these goals must be considered together, as all are essential. A system will not be used if users become frustrated after five minutes of usage (usability) even if it's been shown to aid the user in getting work done in a new way. A business will not adopt a system that is incredibly easy to use but decreases productivity (performance).

Agenda for the course

• Morning I

- Review of technology
- Basic 3D interaction techniques

• Morning II

- Non-isomorphic techniques
- 3D metaphors and strategies for design
- Adapting 2D interfaces

• Afternoon I

- 3D interaction in non-immersive environments
- Evaluation of 3D interfaces

• Afternoon II

- Using 3D interfaces in industry
- Panel discussion

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

See the detailed schedule at the front of these course notes.

Speakers

- Joseph LaViola – Brown University
- Mark Mine – Walt Disney Interactive
- Ivan Poupyrev – Sony Computer Science Labs
- Doug Bowman – Virginia Tech

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

See the speaker bios in the front of these course notes.

Your participation

- **Keep notes on issues to raise at the afternoon panel**
- **Speakers available for questions during breaks**
- **Please fill out the evaluation form**
- **After the conference – via email**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

It is difficult in a course of this size to have interactions between the speakers and the audience, and most of the content here will be presented in lecture format. However, we do value your participation.

The main way you can participate is during the panel session at the end of the day. We encourage you to note questions you would like to raise during the panel as the course progresses.

As often as possible, we will leave time at the end of talks for questions, and speakers will be available during the breaks for more extensive discussions.

We also want your feedback – please fill out the evaluation form.

After the conference, feel free to email us:

bowman@vt.edu

jjl@cs.brown.edu

mine@wdi.disney.com

poup@csl.sony.co.jp

Resources

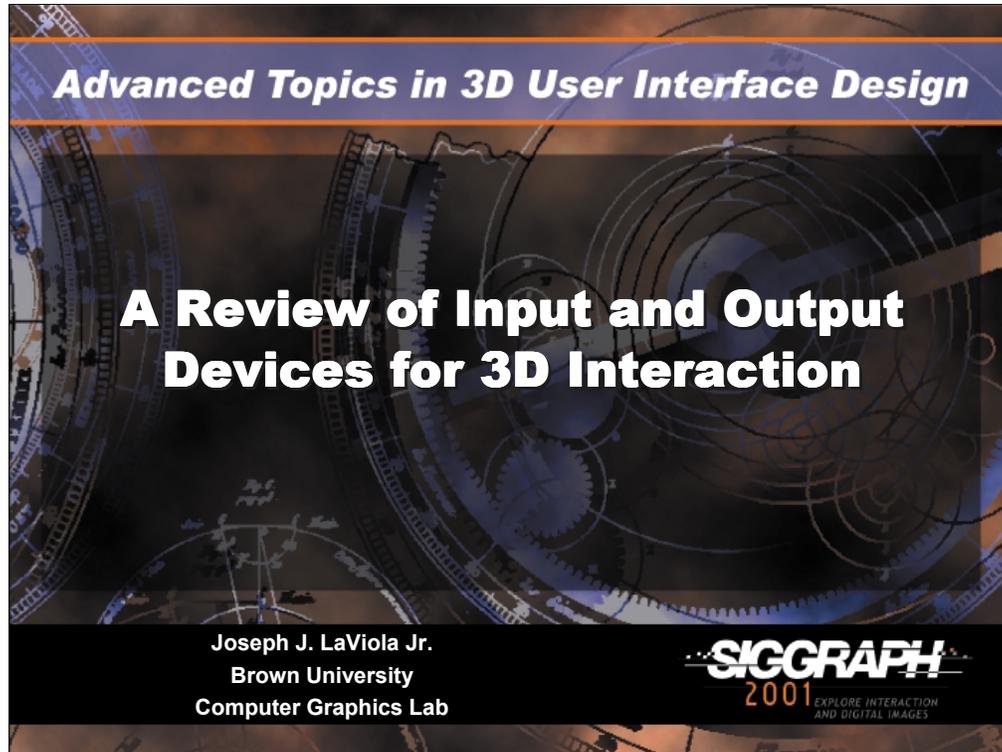
- 3DUI mailing list – email poup@csl.sony.co.jp to join
- 3DUI annotated bibliography
- Course notes
 - Reprints of important papers
 - Extensive notes expanding on the slides



We invite those interested in pursuing this topic further to join the 3DUI mailing list, a list devoted to the discussion of 3D user interfaces and interaction. The list currently has over 100 members from around the world. Email Ivan Poupyrev to join. Also see the 3DUI home page at:

<http://www.mic.atr.co.jp/~poup/3dui.html>

We have included in these course notes the updated 3DUI annotated bibliography, an invaluable resource for researchers in this field. It is also available online. The course notes also include reprints of relevant articles, and the notes you are currently reading, which give additional information about almost all of the topics that will be covered in the course.



A Review of Input and Output Devices for 3D Interaction

Joseph J LaViola Jr

Ph.D Candidate

Brown University,

Department of Computer Science

Providence, Rhode Island

Lead Consultant and Founder

JJL Interface Consultants, Inc.

http: <http://www.cs.brown.edu/people/jjl>

email: jjl@cs.brown.edu

Goals and Motivation

- **Provide practical introduction to the I/O devices used in 3D interfaces**
- **Examine common and state of the art I/O devices**
 - look for general trends
 - spark creativity
- **Advantages and disadvantages**
- **Discuss how different I/O devices affect interface design**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

In this lecture we will discuss the various input and output devices that are used in 3D user interfaces and virtual environment applications.

Lecture Outline

- **Output devices**
 - visual displays
 - audio output
 - olfactory output
 - tactile and haptic output
- **Input devices**
 - discrete event devices
 - continuous event devices
 - combination devices
 - speech input

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The first part of the lecture will describe a number of output devices that stimulate the human visual, auditory, haptic, and tactile systems. In the second part of the lecture, we will look at the many different ways a user can interface to a 3D world. With each device, we will discuss the advantages, disadvantages, and its effects on interface design.

Visual Display Technology

- Two important questions
- How does the light get produced?
- What geometrical surface does the light get displayed on?
- Other criteria
 - FOV
 - ergonomics

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Visual display systems for virtual reality and other 3D applications have two important and interrelated components. The first is the technology underlying how the light we see gets produced; the second is the type and geometrical form of surface on which this light gets displayed. Other criteria for thinking about visual display systems include field of view (FOV) and ergonomics.

Light Producing Technology

- CRT
- LCD
- Digital Light Projectors
- Grating Light Valve Technology
- BlackScreen Technology
- Laser

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

A number of different methods exist for producing the light displayed on a geometrical surface. While Cathode Ray Tubes (CRTs) and Liquid Crystal Display (LCD) panels and projectors are currently the norm in today's marketplace, newer light-producing techniques are emerging. Texas Instrument's Digital Micromirror Device (DMD) is currently available in digital light projectors. The DMD is a thumbnail-size semiconductor light switch which consists of an array of thousands of microscopic sized mirrors, each mounted on a hinge structure so that it can be individually tilted back and forth. When a lamp and projection lens are positioned in the right places, DLP processes the input video signal and tilts the mirrors to generate a digital image.

Silicon Light's Grating Light Valve technology is a micromechanical phase grating which provides controlled diffraction of incident light to produce light or dark pixels in a display system. Their approach can be used to build a 10-bit-per-pixel, high-resolution display compared with 8-bit-per-pixel LCD display. There is hope that this kind of technology may be commoditized for personal displays.

-continued on the next page

Jenmar Visual System's BlackScreen Technology (used in the ActiveSpaces telecollaboration project of Argonne National Laboratories) captures image light into a matrix of optical beads, which focus it and pass it through a black layer into a clear substrate. From there it passes into the viewing area. This screen material presents a black level undegraded by ambient light, making it ideal for use with high-luminosity projection sources and nonplanar tiled displays such as caves.

Finally, laser light is another approach to light production which projects light directly onto the retina. See the slide on Virtual Retinal Displays later in the lecture.

References:

www.dlp.com

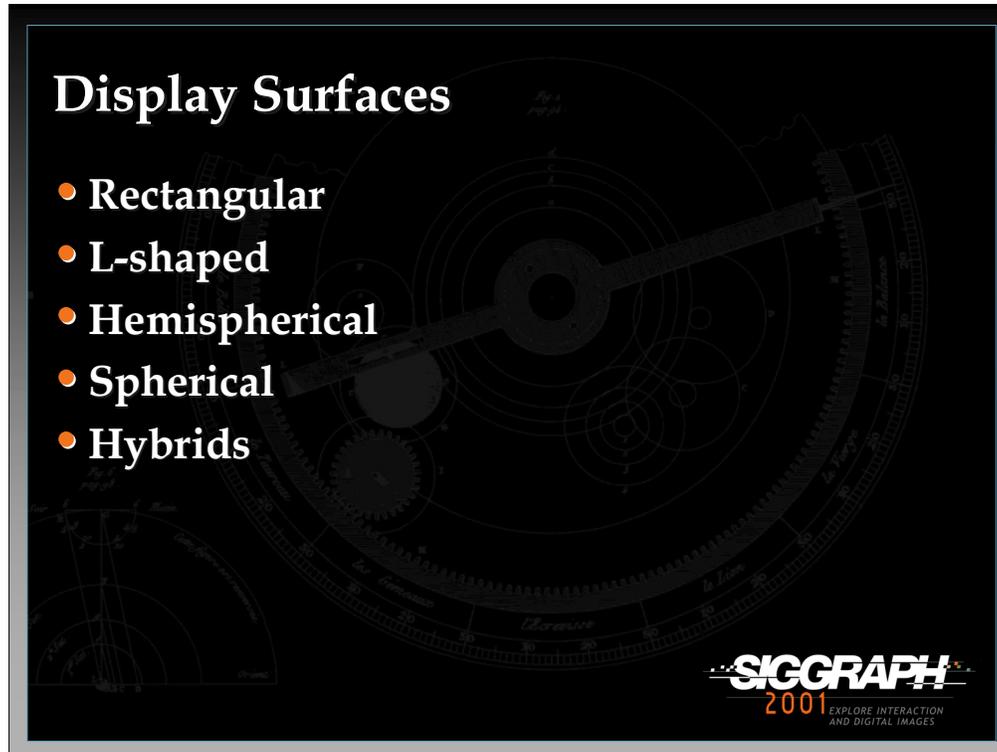
www.siliconlight.com

www.jenmarvs.com

www.mvis.com

www.3d-perception.com

Yoder, Lars. "The Digital Display Technology of the Future" INFOCOMM'97, June 1997.



Unfortunately, no “one size fits all” display surfaces exist for virtual reality and 3D applications. Rather, many different kinds offer advantages and disadvantages. Choosing an appropriate display surface depends on the application, tasks required, target audience, financial and human resources available, and so on. In addition to traditional rectangular display surfaces, more interesting display geometries are starting to affordably emerge including hemispherical and spherical displays and those which combine different geometries together.

Display Device Examples

- HMDs and BOOMs
- SSVR (Cave)
- Workbenches
- conCAVE
- VisionStation
- CyberSphere
- Virtual Retinal Display
- Tiled-Wall Display
- Auto Stereoscopic

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

This slide shows a representative sample of the many visual display devices that exist either in the research lab or in the industrial marketplace. We will look at each example in turn and examine how these devices affect 3D interface design.

HMDs and BOOMs



One of the most common display devices used for virtual environment applications is the head mounted display (HMD). With a tracking device attached to the device, it produces a stereoscopic view that moves relative to the user's head position and orientation. Although traditionally the user cannot naturally see the real world, cameras are sometimes mounted on the HMD which allows it to display both real world video and graphical objects. In addition, some HMDs offer see-through options. This type of technology is used in augmented reality systems.

Since each eye is presented with one screen, HMDs allow for good stereoscopic viewing. These two screens are very close to the user's eyes (1 to 2 inches). As a result, all viewable objects are behind the screen so any object clipping will appear to the user as being outside his/her field of view. A big disadvantage of HMDs is that can get heavy very quickly and, unfortunately, the higher the HMD's quality, the heavier it usually is. Although HMDs are still popular in many VR labs and entertainment centers, researchers and practitioners are rapidly moving towards projection-based display devices especially when high-resolution graphics are required.

-continued on the next page

Since the real world is completely blocked out of the user's view, interaction while wearing an HMD requires the user to have some type of graphical representation of either one or both hands or the input device used. These graphical representations can be as simple as a cube or as complicated as a hand model containing 50000 or more polygons. HMDs also put a strain on the types of input devices that can be used since the user cannot physically see the device in order to use it.

The arm mounted display shown in the picture on the right is called a BOOM developed by Fakespace. It has a counter weight on the the opposite side of the display to make the device easier to manipulate. The device also uses mechanical tracking technology to track the user's head position and orientation. The latest version of the BOOM supports resolutions of 1280x1024 pixels per eye which is better than most average quality HMDs. Since the user does not have to wear the device, it is easy to operate and allows for different users to trade places quickly. As with the HMD, providing one screen per eye allows for good stereo quality. Since the BOOM is physically attached to a large stand, the user's movement is limited. Users can move in about a six foot diameter around the center of the stand. Another disadvantage with the BOOM is the user has to have at least one hand on the device which can limit various types of two-handed interaction.

References:

www.nvis.com

www.virtualresearch.com

www.stereo3d.com

www.fakespace.com

Surround Screen VR



The first surround screen virtual reality system was developed by Carolina Cruz-Neira at the Electronic Visualization Laboratory at the University of Illinois at Chicago. This system was called the CAVE. Today, the term CAVE is copyrighted by Pyramid Systems (now a part of Fakespace). So the general term for such a device is a surround screen environment. These systems also go by other names such as the C2, C6, and TAN Cube and can have anywhere from three to six screens.

The figure in the upper right corner of the slide is called a Computer-driven Upper Body Environment (CUBE). It is a 360° display environment composed of four 32"X28" rear-projected Plexiglas screens. Guests stand inside the CUBE, which is suspended from the ceiling, and physically turn around to view the screen surfaces. The screens are approximately 1' from a guest's face and extend down to his or her midsection. It was developed at the Entertainment Technology Center at Carnegie Mellon University and represents a small-personalized version of a SSVE.

The figure in the lower right corner of the slide shows the RAVE, a reconfigurable advanced visualization environment developed by Fakespace. It is designed to be a flexible display device which can be used as a 30 foot flat wall, a 30 foot variable angle immersive theatre, a Cave-like three wall and floor immersive environment, an L-shaped cove with separate 10 foot wall, and three separate 10 foot review walls.

-continued on the next page

There are a number of advantages to using an SSVR system. They provide high resolution and a large FOV. Users only need a pair of light weight shutter glasses for stereo viewing and have the freedom to move about the device. Additionally, real and virtual objects can be mixed in the environment and a group of people can inhabit the space simultaneously.

One of the biggest disadvantages of SSVR systems is the fact that they are so expensive and require such a large amount of physical space. Another problem with an SSVR system, as well as any projection-based display system, is stereo viewing can be problematic. When the user gets close to the display or when objects appear to be right in front of the user, it becomes more and more difficult to fuse the two images together. Eye strain is a common problem in these situations. Finally, even though multiple users can inhabit the space at one time, due to technological limitations, no more than two can be head-tracked.

Although physical objects do not have to be represented as graphical objects in SSVR systems, an important issue arises when a physical object passes in front of graphical objects that should appear in front of the said physical object. This is a common problem with any projection-based display device and can hinder the immersive experience.

In most cases the user wears a pair of shutter glasses for stereo viewing. These glasses are synched to flicker at a rate equal to the refresh rate of the graphics engine. These signals are sent to the glasses by infrared signal. So, if the signal is blocked, the shutter glasses will stop working and the stereo effect will be disrupted. As a general guideline, it is a good idea to never have the user move his/her hands or other physical objects in the line of sight of the glasses and emitters.

References:

Cruz-Neira, Carolina, Daniel Sandin, and Thomas Defanti. "Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE" SIGGRAPH'93, 135-142.

<http://www.etc.cmu.edu/projects/cube/index.html>

www.fakespace.com

www.mechdyne.com

www.tan.de

WorkBenches



One of the newest types of display devices is the projection-based drafting table. These devices are usually single screen and go by many different names such as Fakespace's Immersadesk and Immersive WorkBench and VersaBench (pictured on the left), the Barco Baron, and the ITI VisionMaker Digital Desk. In some cases just a single vertical screen is used. The second picture to the left shows Fakespace's Mini Workbench. A pressure sensitive display surface for 2D input is an optional feature with the Workbench. The TAN Holobench, shown in the two pictures on the right, is an L-shaped desk which provides a holographic impression to the user since objects appear to be raised above the it.

In general, workbenches provide high resolution displays, make for an intuitive display for certain types of applications (i.e 3D modeling and drafting, virtual surgery), and can be shared by several users. However, due to technological limitations, at most two users can be head tracked and these devices suffer from the same stereo problems that all rear-projected devices do.

References:

www.iti-world.com

www.barco.com

www.fakespace.com

Fakespace conCAVE

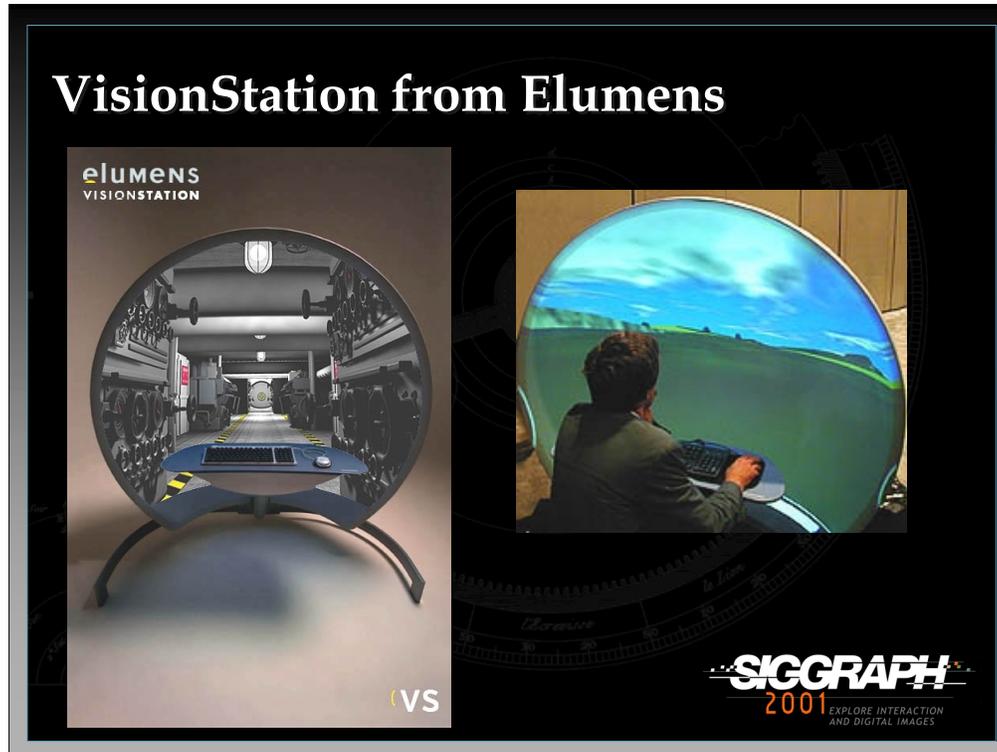


The conCAVE is a rather unique display device in that it combines flat, cylindrical, and spherical display surfaces to form one single device. The conCAVE creates spatially correct 3D “tunnel-view” images of volumetric data that extend from floor to ceiling and side to side. 3D perspective views are depth-enhanced, generating a sense of stereoscopic imagery without the need for special shutter glasses. The conCAVE also has a simple, pull down flat screen in front of the device that creates a large display for standard images such as maps, cross sections, spread-sheets or presentations. From an interface perspective, the device provides the user with both 3D interaction using a tracked paddle and 2D interaction using the virtual buttons on the front of the flat display surface.

References:

www.fakespacesystems.com

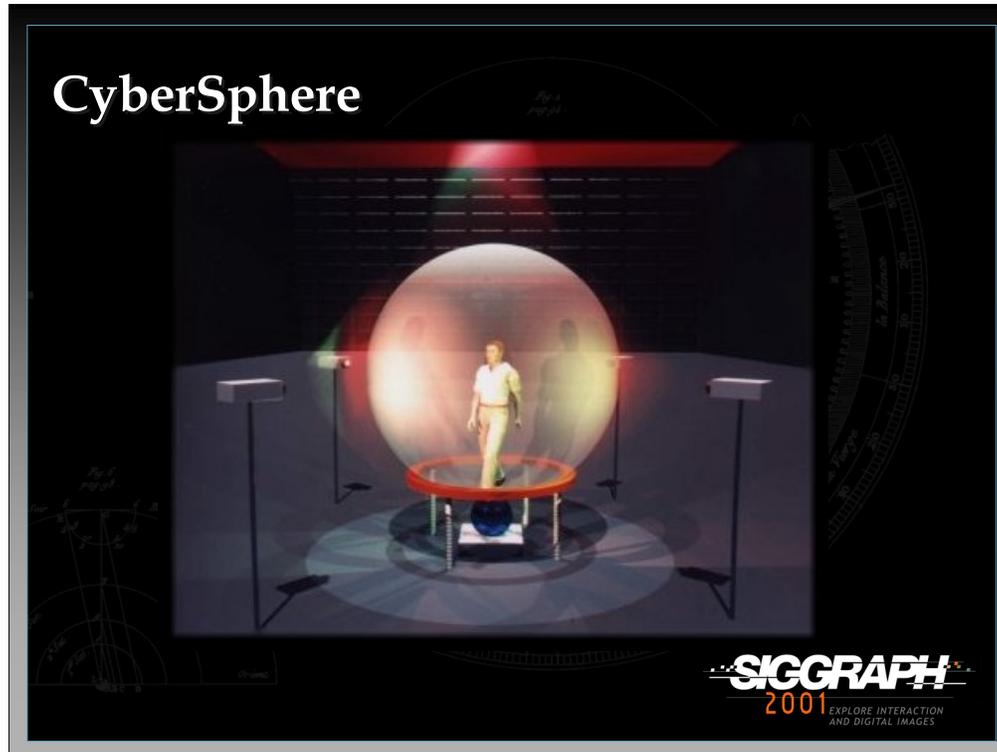
VisionStation from Elumens



The VisionStation is a personalized device that uses a hemispherical front-projected display surface. It uses special proprietary software and optics for the projection lens to display images in a 180 by 180 degree field of view. The user sits in front of a small table and can interact with 3D applications using keyboard and mouse or 3D input devices. One of the major problems with this device is that it is front-projected which means 3D interaction is limited since moving too close to the display surface will cast shadows on the screen.

References:

www.elumens.com

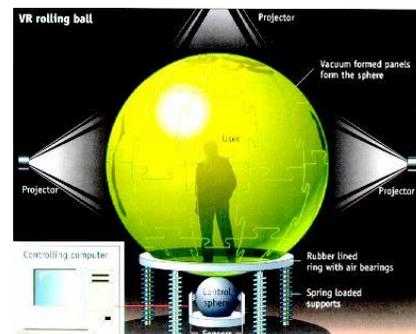


The CyberSphere is a fully spherical immersive display device prototype created by VR Systems UK and the Warwick Manufacturing Group. The system uses a large, hollow, translucent sphere (3.5 meters in diameter) supported by a low pressure cushion of air. The air cushion enables the sphere to rotate in any direction. A single user is able to enter the sphere using a closable entry hatch. Once inside, walking movements cause the large sphere to rotate. This rotational movement is transferred to a smaller secondary sphere, which is supported by means of a ring mounted upon a platform. Rotational movement of the smaller sphere is measured with rotation sensors, pushed against the circumference of the sphere with spring loaded supports. The rotation sensors send signals to the computer which update the projected images in order to give the user the illusion of walking freely through the virtual environment.

The device is still in the prototype stage and is not commercially available.

References:

www.ndirect.co.uk/~vr-systems/sphere1.htm



Virtual Retinal Display



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The Virtual Retinal Display (VRD) was invented at the Human Interface Technology Lab in 1991. It is based on the idea that images can be directly displayed onto the retina. With a VRD, a photon source is used to generate a coherent beam of light which allows the system to draw a diffraction limited spot on the retina. The light beam is intensity modulated to match the intensity of the image being rendered. The beam is then scanned to place each image point, or pixel at the proper position of the retina. VRDs are commercially available from Microvision and are used in augmented reality systems. For more details see the references below.

Reference:

www.hitl.washington.edu/research/vrd/

www.mvis.com

Tiled Wall Displays



Tiled display surfaces, which combine many display surfaces and light producing devices, are becoming increasingly popular for a number of 3D and virtual environment applications. Tiled displays offer greater image fidelity than other immersive and desktop displays due to an increased number of pixels displayed over an area that fills most of a user's or group of user's FOV. The display in the figure is the Scalable Display Wall developed at Princeton University. It has a resolution of 8192 by 3064 pixels and is 18 feet long and 8 feet high. Although an intriguing display device, the tiled wall display has a number of research challenges including hardware setup, maintaining projector calibration and seamless imaging.

References:

<http://www.cs.princeton.edu/omnimedia/index.html>

“Special Issue on Large Wall Displays”, IEEE Computer Graphics and Applications, Vol. 20, No. 4, July/Aug. 2000.

Auto Stereoscopic Displays



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

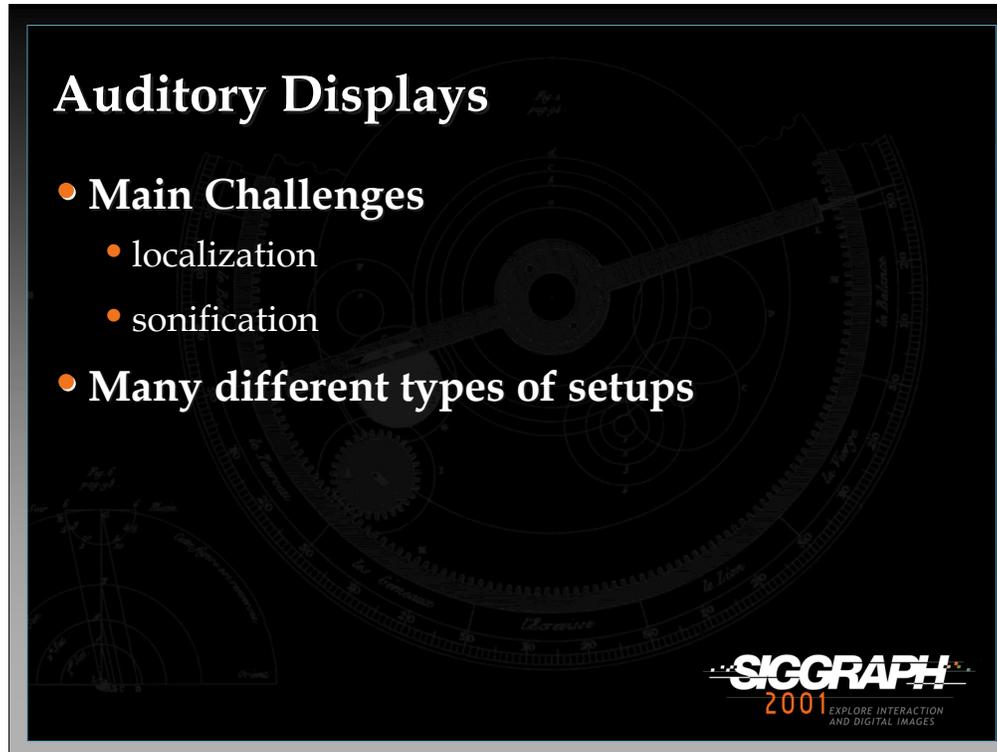
Other visual output devices use lenticular, volumetric, and holographic display technology. Most of these technologies are expensive and in the early stages of development so they are rarely utilized in mainstream 3D interfaces. However, once these technologies become more affordable and the technology has progressed sufficiently, a number of interesting interface issues will arise.

The picture on the left shows a 14 inch lenticular display developed at the Heinrich-Hertz Institute for Communication Technology in Berlin, Germany.

The picture on the right show a lenticular display developed at the Dresden University of Technology.

References:

<http://at.hhi.de>



There are two different ways, localization and sonification, in which sound can be used as an output medium in virtual environment applications. In localization, the goal is to generate three dimensional sound. In sonification, the goal is to turn certain types of information into sounds.

There are a number of different ways in which an auditory system can be setup. A simple setup is to use stereo head phones. However, this restricts usage to only one person at a time. Another setup is to place speakers in certain logistic areas around the environment. This setup allows for more than one user to take part in the experience but is somewhat more complicated to setup and write software for.

Reference:

Begault, Durand R. 3D Sound For Virtual Reality and Multimedia. Academic Press, 1994.

Auditory Output – Interface Design

- If used properly can be a powerful tool
- Tells user something important is happening and where to look for it
- Provides sensory substitution

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Auditory output can be very powerful when applied correctly in 3D virtual environments. It is especially useful in collaborative applications where participants can get a sense for where others are in the environment. It also can be used for sensory substitution which is important when, for example, no haptic or tactile feedback is present. A sound could substitute the feel of a button press or the moving of an object in the virtual space.

Olfactory Output

- **Least developed area**
 - maybe for good reason!
- **Have practical applications**
 - fire fighting
 - surgical training
- **Number of practical problems**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Olfactory interfaces are one of the least developed areas the virtual reality and 3D applications. There are a number of interesting design considerations when dealing with olfactory output such as odor storage and display as well as cleaning the air input and controlling the breathing space for the individual.

References:

www.hitl.washington.edu/people/TFurness/courses/inde543/reports/3doc/

Youngblut, Christine, Johnson, Rob E., Nash, Sarah H., Weinclaw, Ruth A., Will, Craig A., Review of Virtual Environment Interface Technology IDA Paper P-3186. Chapter 8, p. 209-216, <http://www.hitl.washington.edu/scivw/IDA/>.

Dinh, H.Q., N. Walker, L.F. Hodges, C. Song, and A. Kobayashi, "Evaluating the Importance of Multi-sensory Input on Memory and the Sense of Presence in Virtual Environments", In IEEE Virtual Reality'99, 222-228, 1999.

Haptic and Tactile Feedback (1)

- “For every action there is an equal and opposite reaction”
 - Sir Isaac Newton
- Main forms of feedback
 - ground referenced
 - body referenced
 - tactile
 - dermal tactile



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Haptics represents a critical component in virtual environment interaction. Allowing a user to touch and feel in the virtual world in the same way that they do in the physical world is extremely powerful. Unfortunately, haptic and tactile output device research is still in its early stages.

There are essentially four different methods in which haptic and tactile feedback is generated. The first method is ground-referenced feedback which creates a physical link between the user and ground with the feedback relative to a single contact point. An example shown in the bottom picture is Virtual Technologies' CyberForce. The second method is body-referenced feedback which places a device on some part of the user's body. An example of a body-referenced haptic device is Virtual Technologies' CyberGrasp which is shown in the top picture. The third method for generating feedback is tactile which uses some type of oscillatory or vibrating device to stimulate the user's tactile sense. Finally, the last method of generating feedback is via dermal tactile which stimulates the user's nerves in the fingertips.

References:

www.sensable.com

www.virtex.com

Haptic and Tactile Feedback (2)

- Motionware device
- Provides vestibular stimulation
- Sends signals to the 8th cranial nerve
- Gives user a sense of motion



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Another type of tactile feedback device is Motionware being developed by Virtual Motion. Motionware sends electrical current to the 8th cranial nerve located behind the wearer's ear. Sending these electrical signals to the 8th cranial nerve provides the user with vestibular stimulation which can mimic the sense of motion. The device will be priced at around \$100.

References:

www.virtual-motion.com

Haptics – Interface Design

- Useful for object manipulation
- Problem with these devices is they are very intimidating
- Mimic real world interaction



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Haptic feedback from devices like the CyberGrasp are very good for grabbing objects and moving them around and they can provide a limited form of real world interaction. The main problem with these devices is that they are somewhat intimidating to the user. People are commonly afraid to put these devices on and once they do they're afraid they'll break them or get injured. The picture shows the Phantom 3.0 from Sensable Technologies. This device is less intrusive than the CyberGrasp.

References:

Burdea, Grigore C. Force and Touch Feedback for Virtual Reality. Wiley Interscience, 1996.

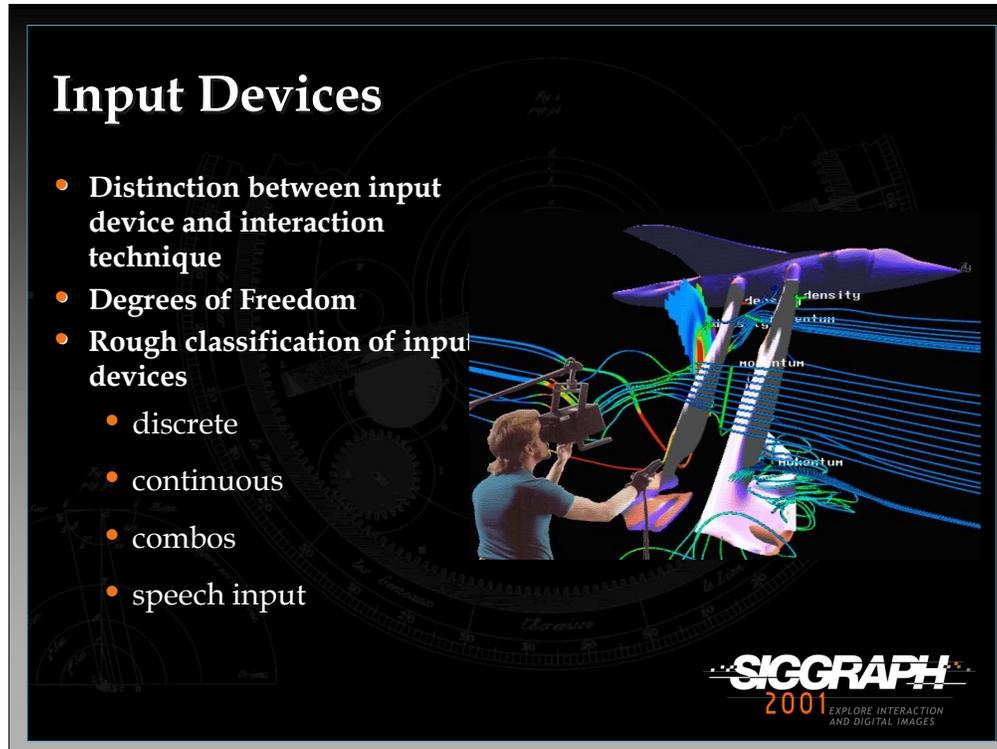
Other Haptic Devices



There are many different haptic devices that are being developed in research labs around the world. The slide shows from top to bottom, left to right the Pneumatic Master Arm from Southern Methodist University, a 5 DOF haptic device from the University of Colorado, a magnetic levitation haptic interface from Carnegie Mellon University, and a tactile display from the Karlsruhe Research Center in Germany.

References:

haptic.mech.northwestern.edu



There is a distinction that must be made when we talk about input devices and interaction techniques. Input devices are just the physical tools that are used to implement various interaction techniques. In general, many different interface techniques can be mapped onto any given input device. The question is how natural, efficient, and appropriate a given input device will work with a given technique.

When talking about input devices it is convenient to talk about the degrees of freedom (DOF) that an input device has. For example, a device such as a tracker generally produces 3 position values and 3 orientation values for a total of 6 DOF. For the most part, a device with a smaller number of DOF can be used to emulate a device with a higher DOF with the addition of buttons or modifier keys.

See the papers by Shumin Zhai in the papers section of the course notes for a series of experiments that evaluate a number of the input devices presented in this part of the lecture.

Discrete Input Devices

- Generate one event at a time based on the user

- Examples

- Keyboard
- Pinch Glove (see picture)
- Interaction Slippers
- Painting Table



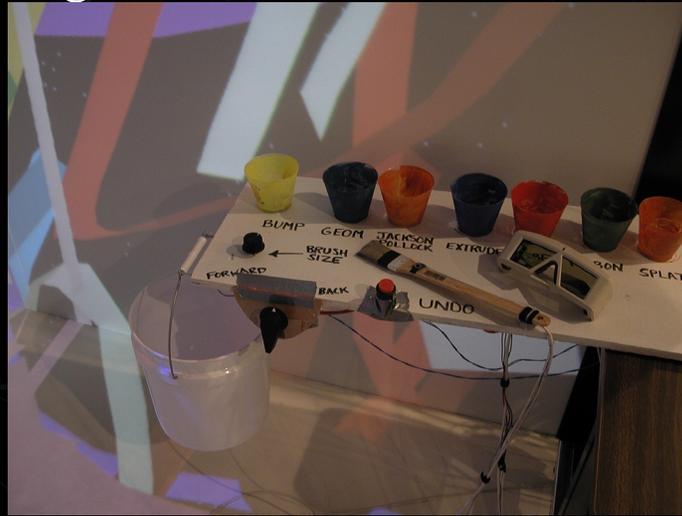
SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Discrete input devices simply generate one event at a time based on the user. In other words, when the user presses a button an event is generated which is usually a boolean value stating whether the button was pressed down or released. The keyboard is an obvious example of a discrete input device.

The Pinch Glove system developed by Fakespace is another example of a discrete input device. These gloves had a conductive material at each of the fingertips so that when the user pinches two fingers together a electrical contact is made which generates a boolean value. There are many different pinching combinations that can be made which allows for a significant amount of input device to task mappings.

The Interaction Slippers are a custom made device which allows users to perform toe and heel tapping for invoking commands. The slippers use conductive cloth contacts and a Magellan Trackman Live! wireless mouse. See the paper, “Hands-Free Multi-Scale Navigation in Virtual Environments”, in the papers section of the course notes for more details.

Painting Table



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The Painting Table is another example of a discrete input device that is used in the CavePainting application, a system for painting 3D scenes in a virtual environment. The device uses a set of conductive cloth contacts as well as traditional buttons and digital sliders. Users can dip the paint brush prop into the colored cups to change brush strokes. The bucket is used to throw paint around the virtual canvas.

References:

Keefe, D., Acevedo, D., Moscovich, T., Laidlaw, D., and LaViola, J.
“CavePainting: A Fully Immersive 3D Artistic Medium and Interactive Experience”, Proceedings of the 2001 Symposium on Interactive 3D Graphics, 85-93, 2001.

Continuous Input Devices

- **Continuously generate events in isolation or in response to user action**

- **Examples**

- trackers
- datagloves
- bioelectric control
- body sensing devices
- Cyberlink



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Continuous input devices generate a continual stream of events in isolation (no user manipulation) or in response to user action. For example, a tracker is a device which will continually output position and orientation records even if the device is not moving. These types of devices are important when we want to know where something is in the virtual space and we do not want to have to keep asking for it. A perfect example of this is head tracking. Two of the most common continuous devices are trackers and datagloves.

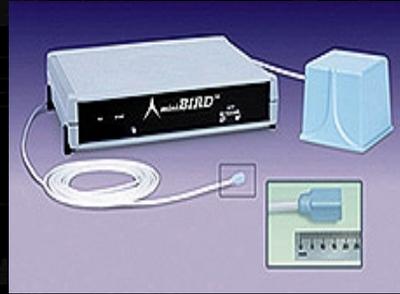
Another type of continuous input device is the Cyberlink, a brain-body actuated control technology that combines eye-movement, facial muscle, and brain wave bio-potentials to generate input signals. The Cyberlink has three sensors in a headband and its interface unit amplifies and translates the brain wave, facial muscle and eye-movement data into separate frequencies and transmits them to an PC serial port. The Cyberlink software processes and displays these frequencies and 10 continuous command signals called Brainfingers.

References:

www.brainfingers.com

Trackers

- **Goals and importance**
 - provide correct viewing perspective
 - correspondence between physical and virtual worlds
- **Types of trackers**
 - magnetic
 - mechanical
 - acoustic
 - inertial
 - vision/camera
 - hybrids



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

One of the most important aspects of 3D interaction in virtual worlds is providing a correspondence between the physical and virtual environments. As a result, having accurate tracking is extremely important to making the VE usable. Currently there are a number of different tracking technologies in the marketplace. The different types are shown in the slide.

Magnetic tracking uses a transmitting device that emits a low frequency magnetic field that a small sensor, the receiver, uses to determine its position and orientation relative to a magnetic source. These trackers can use extended range transmitters which increase the range of the device from around an 8 foot radius to anywhere from a 15 to 30 foot radius. The tracker shown in the picture is called the Ascension MiniBird. It uses a smaller emitter and receivers and has better accuracy than the regular system. However it's range is limited to about a 4 foot radius. It is primarily used in medical applications where range of the device is not a factor.

Mechanical trackers have a rigid structure with a number of joints. One end is fixed in place while the other is attached to the object to be tracked (usually the user's head). The joint angles are used to obtain position and orientation records. The Fakespace BOOM uses this type of tracking technology.

-continued on the next page

Acoustic tracking devices use high frequency sound emitted from a source component that is placed on the hand or object to be tracked. Microphones placed in the environment receive ultrasonic pings from the source components to determine their location and orientation. In most cases, the microphones are placed in a triangular fashion and this region determines the area of tracked space. One of the most interesting problems with this type of tracking is that certain noises such as jingling keys or a ringing phone will interfere with the device.

Inertial tracking systems use a variety of inertial measurement devices such as gyroscopes, servo accelerometers, and micro-machined quartz tuning forks. Since the tracking system is in the sensor, range is limited to the length of the cord which attaches the sensor to the electronics box. Two of the big limitations of these devices is that they only track orientation and are subject to error accumulation. The InterSense IS300 handles error accumulation by using a gravitometer and compass measurements to prevent accumulation of gyroscopic drift and also uses motion prediction algorithms to predict motion up to 50 milliseconds into the future.

Camera/vision based tracking take one or more cameras and places them in the physical environment. The cameras then grab video of the user or object to be tracked. Usually image processing techniques, such as edge detection algorithms, are used to identify the position and/or orientation of various body parts such as the head and hands. Setting up vision-based tracking systems can be difficult since there are many parameters that must be fixed in order to track the user properly. These parameters include the number of cameras, the placement of the camera, what background (what is in back of the user) is put up, and if the user will be wearing special optical tools such as LEDs or colored gloves to aid in tracking. Ascension's laserBIRD is an example of an optical tracking device. laserBIRD delivers accurate position and orientation tracking without environmental interference or distortion. Its miniaturized scanner reflects laser beams throughout the work space. Each sensor, attached to a tracked object, instantly picks up the laser beams. Signals are then directed back to the scanner's DSP electronics for processing and transmission to a host PC or workstation. Other vision-based approaches include the UNC HighBall which uses LED beacons mounted on the ceiling.

-continued on the next page

Hybrid trackers attempt to put more than one tracking technology together to help increase accuracy, reduce latency, and, in general, provide a better virtual environment experience. An example is the InterSense IS600. It combines inertial and ultrasonic tracking technologies which enables the device to attain 6 DOF. The major difficulty with hybrid trackers is that the more components added to the system, the more complex the device becomes.

Other types of hybrid tracking include the combination of video cameras and structured digital light projectors. Combining these two technologies allow for the capture of depth, color, and surface reflectance information for objects and participants in the environment. This approach is currently being used at the University of North Carolina, Chapel Hill in their Office of the Future project as well as in the National Tele-Immersion Initiative. The picture shows two user collaborating in two remote locations.



References:

www.ascension-tech.com

www.polhemus.com

www.isense.com

www.3rd-tech.com (Commercial version of the UNC HighBall Tracker)

Raskar, Ramesh, Welch, Greg, et al. "The Office of the Future: A Unified Approach to Image-Based Modeling and Spatially Immersive Displays" SIGGRAPH '98, ACM Press, 179-188.

Amela Sadagic et. al., "National Tele-Immersion Initiative: Towards Compelling Tele-Immersive Collaborative Environments", Medicine meets Virtual Reality 2001 conference, January 24-27, 2001.

Eye Tracking



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Eye tracking systems provide applications with knowledge of the user's gaze direction. This information opens the door to a number of interesting interaction techniques such as eye directed selection and manipulation. The figure on the left shows the Eyegaze system, a non-intrusive approach which uses an infra-red source that reflects off of the pupil, developed by LC Technologies. The figure on the right shows iView, a head-mounted eye tracking device developed by SensoMotoric Instruments.

References:

www.eyegaze.com

www.smi.de

Data Gloves

- **Used to track the user's finger movements**

- for gesture and posture communication

- **Types**

- CyberGlove
- 5DT Glove 16-W



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Data gloves measure finger movement of the hand by using various kinds of sensor technology. These sensors are embedded in the glove or placed on top of the glove, usually on the back of the hand. The number of sensors in the glove depends on the manufacturer. Virtual Technologies' CyberGlove has either 18 or 22 sensors which can measure at least 2 joints in each finger, wrist roll and yaw, and others. These types of gloves are commonly used for hand gesture and posture recognition which can be applied to a variety of different interface techniques in virtual environments. Fifth Dimension Technologies (5DT) offers gloves that have either 5 sensors, one for each fingertip or 16 sensors, 2 for each finger and abduction between fingers. 5DT also has wireless versions of each glove.

References:

www.virtex.com

www.5dt.com

Bioelectric Control

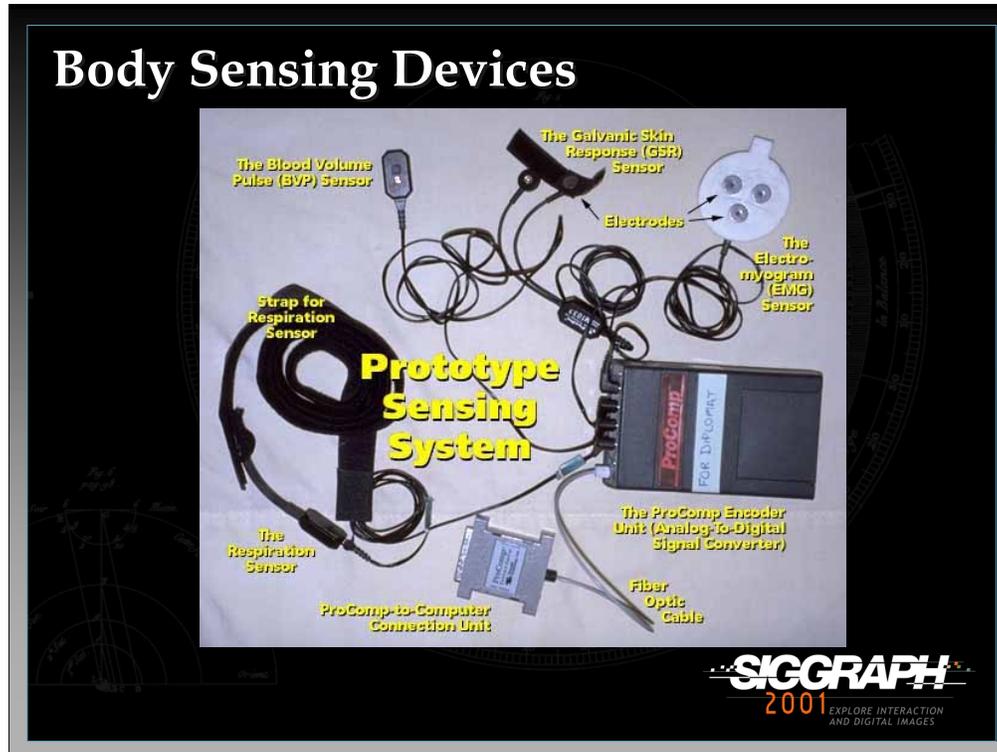


A recent development at NASA Ames Research Center is a bioelectric input device which reads muscle nerve signals emanating from the forearm. These nerve signals are captured by a dry electrode array on the arm. The nerve signals are analyzed using pattern recognition software and then routed through a computer to issue relevant interface commands. The figure on the left shows a user entering numbers on a virtual numeric keypad while the figure on the right shows a user controlling a virtual 757 aircraft.

References:

Jorgensen, Charles, Kevin Wheeler, and Slawomir Stepniewski. Bioelectric Control of a 757 Class High Fidelity Aircraft Simulation, <http://ic.arc.nasa.gov/publications/index.html>, 1999.

Body Sensing Devices



The MIT Media Lab's affective computing group has developed a Prototype Physiological Sensing System which includes a Galvanic Skin Response sensor, a Blood Volume Pulse sensor, a Respiration sensor, and an Electromyogram. By using this prototype, interface developers can monitor a user's emotional state to dynamically modify an application's interface to better fit the user's needs.

References:

<http://www.media.mit.edu/affect/>

Combination/Hybrid Devices (1)

- Devices have the ability to generate both discrete and continuous events
- Classic example - Mouse
- Joysticks (pictured)
- Tablets



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

A combination/hybrid input device combines both discrete and continuous event generating devices to form a single device that is more flexible. Two of the most common hybrid devices are the joystick and mouse. Another device in this category is the pen-based tablet. Pen-based tablets are becoming more and more popular in virtual environment applications because they give the user the ability to interact in 2D which provides a useful combination in certain interfaces. The figure shows the SpaceStick developed by MUSE Virtual Presence.

References:

www.vrweb.com

www.wacom.com

Combination/Hybrid Devices (2)

- Space Mouse (Magellan)
- Ring Mouse
- Fly Mouse
- Isometric Devices
 - Spaceball
 - SpaceOrb



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The Space Mouse (Magellan) is a 6 DOF input device originally designed for telerobotic manipulation. Slight pressure of the fingers onto the cap of the Magellan generates small deflections in X, Y, and Z, which can move objects in 3D space. With slight twists of the cap, rotational motions are generated. It also has a series of buttons which will generate discrete events. The Ring Mouse (top picture) is a small device worn on the user's finger which uses ultrasonic tracking. It also has two buttons for generating discrete events. The main advantages of this device is that it is wireless and inexpensive. The Fly Mouse is a 3D mouse that also uses ultrasonic tracking. This device has five buttons instead of two and also can be used as a microphone.

Another type of input devices are isometric which have a large spring constant so they cannot be perceptibly moved. Their output varies with the force the user puts on the device. A translation isometric device is pushed while a rotation isometric device is twisted. A problem with these devices is that users may tire quickly from the pressure they must apply in order to use them. The bottom figure is a picture of the SpaceOrb, an isometric device from Labtec priced at approximately forty dollars.

References:

www.spacemouse.com, www.labtec.com, www.pegatech.com
www.qualixdirect.com/html/3d_mouse_and_head_tracker.html

Combination/Hybrid Devices (3)

- BAT
- Wand
- Flex and Pinch
- Lego Interface Toolkit



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The BAT is a device that was developed by Colin Ware in the late 1980's. It essentially is just a tracking device with three buttons attached to it. It's similar to the other 3D mice mentioned in the previous slide except it is rather easy to build one with a few electrical components (provided you have the tracking device). The Wand is a device that is commonly seen in SSVR environments. It is simply a more elegant version of the BAT that is commercially developed. The Flex and Pinch input system is a custom built device which takes the functionality of the Pinch Glove system and combines it with the bend sensing technology of a data glove. The pinch buttons are made from conductive cloth and can be placed anywhere on the bend sensing glove. The Lego Interface Toolkit is a rapid prototyping system for physical interaction devices in immersive environments. It utilizes Lego bricks because they are easily obtained and support a variety of physical configurations.

References:

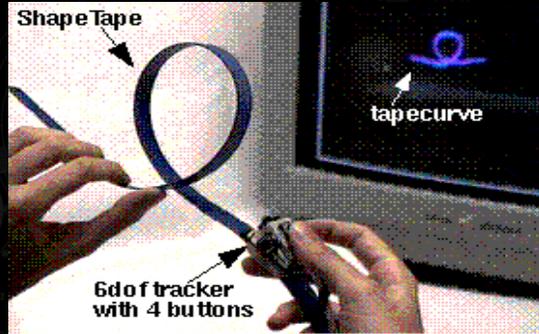
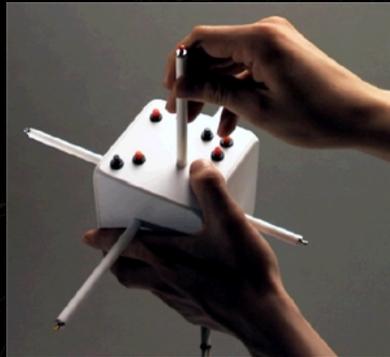
Ware, Colin and Danny R. Jessome. "Using the Bat: A Six Dimensional Mouse for Object Placement." Proceedings of Graphics Interface '88, 119-124.

LaViola, Joseph and Robert Zeleznik. "Flex and Pinch: A Case Study of Whole-Hand Input Design for Virtual Environment Interaction." Proceedings of the IASTED International Conference on Computer Graphics and Imaging '99, 221-225.

Ayers, Matthew and Robert Zeleznik. "The Lego Interface Toolkit." Proceedings of User Interface Software and Technology, 1996, 97-98.

Combination/Hybrid Devices (4)

- ShapeTape
- Cubic Mouse



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

ShapeTape is a continuous bend and twist sensitive strip which encourages two-handed manipulation. A BAT is attached and the tool (shown in the figure on the right) is used for creating and editing curves and surfaces along with camera control and command access. ShapeTape senses bend and twist with two fiber optic sensors at 6cm intervals.

The Cubic Mouse (shown in the figure on the left) is an input device developed at GMD that allows users to intuitively specify three-dimensional coordinates in graphics applications. The device consists of a box with three perpendicular rods passing through the center and buttons for additional input.

References:

Balakrishnan, Ravin, George Fitzmaurice, Gordon Kurtenbach, and Karan Singh. "Exploring Interactive Curve and Surface Manipulation Using a Bend and Twist Sensitive Input Strip" Proceedings of the 1999 Symposium on Interactive 3D Graphics, 111-118, 1999.

Frohlich, Bernd, John Plate. "The Cubic Mouse: A New Device for Three-Dimensional Input", Proceedings of CHI2000, 526-531, 2000.

Speech Input

- Provides complement to other modes of interaction
- Ideal for multimodal interaction

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Speech input provides a nice complement to other input devices. As a result, it is a natural way to combine different modes of input (e.g. multimodal interaction) to form a more cohesive and natural interface. In general, when functioning properly speech input can be a valuable tool in virtual environment applications especially when both of the user's hands are occupied. There are many issues to consider when dealing with speech input besides what speech recognition engine to use. There are tradeoffs that must be made when dealing with speech input. An important issue is where the microphone is to be placed. Ideally, a wide area mike would be best so that the user does not have to wear a headset. Placing such a microphone in the physical environment could be problematic since it might pick up noise from other people or machines in the room. One of the big problems with using speech input is having the computer know when to and not to listen to the user's voice. Often, a user is conversing with a collaborator with no intention of issuing voice commands but the applications "thinks" the user is speaking to it. This misinterpretation can be very problematic. One of the best ways to avoid this problem is to use an implicit or invisible push-to-talk scheme. A push-to-talk scheme lets the user tell the application when he/she is speaking to it or someone else. In order to keep the naturalness of the speech interface, we do not want to have to add to the user's cognitive load. The goal of implicit push-to-talk is to imbed the "push" into existing interaction techniques so the user does not have the burden of remembering to signal the application that a voice command is about to be issued.

References:

LaViola J. Whole-Hand and Speech Input in Virtual Environments, Master's Thesis, Brown University, Dept. of Computer Science, December 1999.

Conclusions

- **Money is a big factor**
- **Think about what interaction techniques are required**
- **Choosing input device restricts the choice of output device**
- **Choosing output device restricts the choice of input device**
- **Creativity is important**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

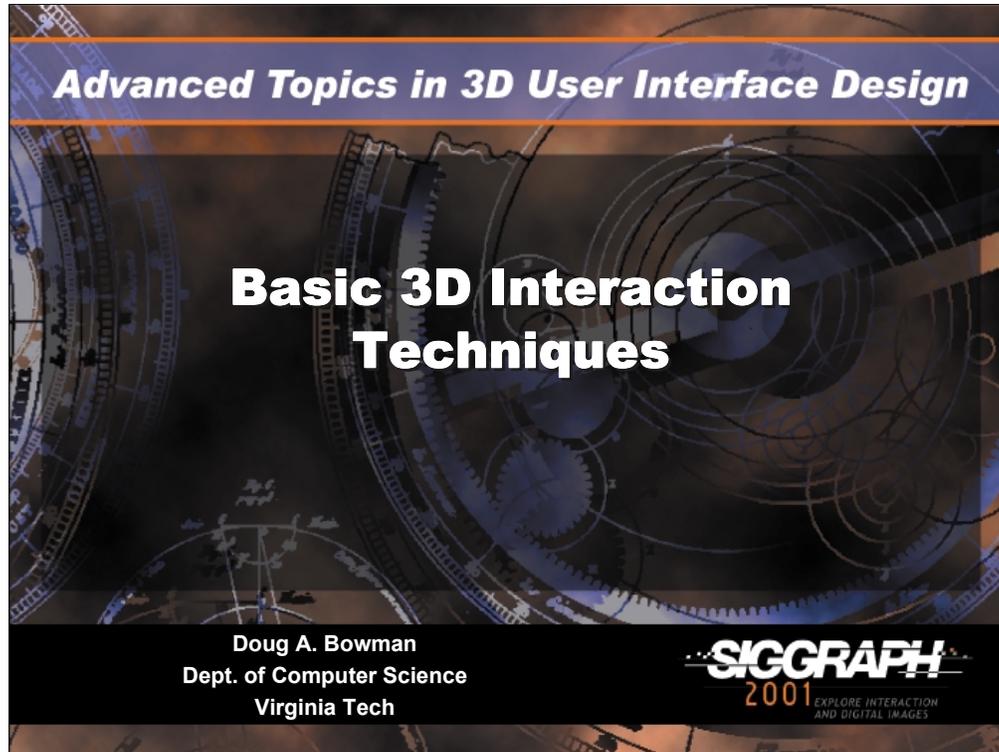
Obviously when choosing input and output devices for creating virtual environment applications and systems, money is a big issue. However, getting the most expensive I/O devices does not necessarily guarantee that the VE will be usable. In general, when selecting I/O device, think about what the user is going to be doing in the VE and what sorts of interaction techniques will be required. At that point, thinking about the physical devices that are best suited for the required techniques.

Finally, none of the input or output devices described in this lecture are perfect. As a result, there is a lot of research left to be done to develop better I/O devices. Creativity is important when thinking about them. If you can't find a commercially available device to suit your needs then build one that will.

General Reference:

Carolina Cruz-Niera, "Applied Virtual Reality." Course #14. Siggraph 1998.

Youngblut, C. R.E. Johnson, S.H. Nash, R.A. Wienclaw, and C.A. Will, "Review of Virtual Environment Interface Technology." Technical Report IDA Paper P-3186, Log:H96-001239. Institute for Defense Analysis. 1996.



Basic 3D interaction techniques

Doug A. Bowman

Dept. of Computer Science (0106)

660 McBryde Hall

Virginia Tech

Blacksburg, VA 24061 USA

Email: bowman@vt.edu

Web: <http://www.cs.vt.edu/~bowman/>

In this lecture, we'll describe several common techniques for basic 3D tasks. All of these techniques were described in last year's course notes, with an emphasis on their advantages and disadvantages, and on their usability characteristics. This year, for the advanced course, we take a different approach. We describe details of the *implementation* of these techniques, including mathematics needed to properly implement them. This information has not been compiled in one place before, to our knowledge, so we hope it will be useful as you implement your own 3D interfaces.

The techniques we describe have mostly been developed in the context of VEs, but many of them are useful in other types of 3D systems as well. We also have a lecture on non-VE interaction later in the course.

Universal 3D interaction tasks

- **Navigation**
 - Travel: motor component of viewpoint motion
 - Wayfinding: cognitive component; decision-making
- **Selection: picking object(s) from a set**
- **Manipulation: modifying object properties (esp. position/orientation)**
- **System control: changing system state or mode**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

We'll be discussing techniques for four basic 3D interaction tasks that are found in most complex 3D applications. Obviously, there are other tasks which are specific to an application domain, but these are some basic building blocks that can often be combined to create a more complex task.

Navigation is the most common VE task, and is actually composed of two tasks. Travel is the motor component of navigation, and just refers to the physical movement from place to place. Wayfinding is the cognitive or decision-making component of navigation, and it asks the questions, "where am I?", "where do I want to go?", "how do I get there?", and so on.

Selection is simply the specification of an object or a set of objects for some purpose. Manipulation refers to the specification of object properties (most often position and orientation, but also other attributes). Selection and manipulation are often used together, but selection may be a stand-alone task. For example, the user may select an object in order to apply a command such as "delete" to that object.

-continued on the next page

System control is the task of changing the system state or the mode of interaction. This is usually done with some type of command to the system (either explicit or implicit). Examples in 2D systems include menus and command-line interfaces. It is often the case that a system control technique is composed of the other three tasks (e.g. a menu command involves selection), but it's also useful to consider it separately since special techniques have been developed for it and it is quite common.

Assumed knowledge

- **Scene graphs / object trees**
 - Parent/child relationships
 - Representation of user
- **Affine transformations with matrices**
 - Translate
 - Rotate
 - Scale
- **This lecture assumes that y is up**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

We are assuming a basic level of knowledge in this lecture, so that we do not have to cover the lowest-level implementation details for these interaction techniques. These concepts should be familiar to anyone who has done graphics programming.

First, we assume you are familiar with hierarchical scene graphs or object trees in 3D graphics systems, including the concept of parent/child relationships between objects, inherited transformations, and the representation of the user in a scene graph.

Second, we assume you are familiar with matrix representations of simple transformations such as translate, rotate, and scale, and with the concept of composite matrices for multiple transformations.

Finally, a note on notation: we assume (for the techniques where it makes a difference) that 'y' is the vertical axis in the world coordinate system. Some systems will use 'z' as the vertical axis.

Mapping between coordinate systems

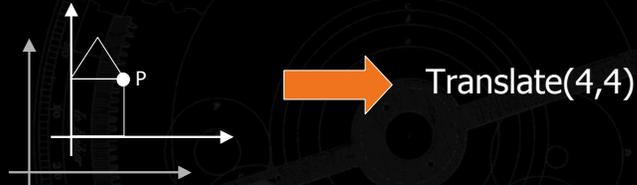
- **Given:**
 - The vertices of an object in CS_2
 - A transformation matrix M that transforms CS_1 to CS_2
- **What are the coordinates of the object's vertices in CS_1 ?**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Before we begin with the techniques, we need to review an important concept. In most graphics systems, objects and their vertices can be described in two ways: their location (position, orientation, and scale) in a fixed world coordinate system, or in a local coordinate system attached to the object. For many techniques, we will need to distinguish between world and local coordinate systems, and in some cases move from one representation to another. Therefore, we discuss here the concept of mapping between coordinate systems.

The problem we want to solve is this: given the vertices of an object in one coordinate system (say its local coordinate system), and a transformation matrix M that transforms another coordinate system (say the world CS) to the first, what are the coordinates of the vertices in the world coordinate system?

Mapping example



Point P is at (2,2) in the transformed CS (CS_2).
Where is it in CS_1 ?

Answer: (6,6)

*Note: $(6,6) = T_{4,4} P$

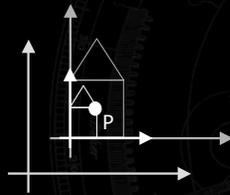
In general, if CS_1 is transformed by a matrix M to form CS_2 , a point P in CS_2 is represented by MP in CS_1

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

This problem is relatively trivial. If the corner of the house is at (2,2) in CS_2 , and the mapping between CS_1 and CS_2 is a translation by (4,4), then the corner of the house is obviously at (6,6) in CS_1 .

Therefore, we can see that if M is the mapping between CS_1 and CS_2 , then $P(CS_1) = MP(CS_2)$.

Another example



Translate(4,4), then
Scale(0.5, 0.5)

Where is P in CS₃?

(2,2)

Where is P in CS₂?

$S_{0.5,0.5}(2,2) = (1,1)$

Where is P in CS₁?

$T_{4,4}(1,1) = (5,5)$

*Note: to go directly from CS₃ to CS₁ we can
calculate $T_{4,4} S_{0.5,0.5}(2,2) = (5,5)$

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

However, the mapping between coordinate systems may be more complicated than a single transformation. This is especially true when using scene graphs, where there may be many objects (and therefore many transformations) between the world coordinate system and the local coordinate system of an object. So, we need a general rule.

We see here that if we do two transformations to go from CS₁ to CS₃, then we can find a point's location in CS₁ by first multiplying the transformation from CS₁ to CS₂ by the transformation from CS₂ to CS₃ (in that order – remember that matrix multiplication is not commutative), then multiplying the result by the point's location in CS₃.

General mapping rule

- If CS_1 is transformed consecutively by M_1, M_2, \dots, M_n to form CS_{n+1} , then a point P in CS_{n+1} is represented by $M_1 M_2 \dots M_n P$ in CS_1 .
- To form the composite transformation between CS's, you postmultiply each successive transformation matrix.

SIGGRAPH
2001 EXPLORE INTERACTION
AND DIGITAL IMAGES

Therefore, the general rule is that we can form a composite transformation matrix between coordinate systems by postmultiplying each successive transformation matrix. (This is the opposite of the procedure if you are transforming the vertices themselves, instead of the CS). The OpenGL graphics package implicitly encourages this type of transformation through its use of transformation matrix stacks.

Remember, this general rule can be applied whenever a technique requires you to do a transformation between coordinate systems.

Implementation issues for travel techniques

- Velocity / acceleration control
- Is world rotation necessary?
- Constrained motion
 - Constant height
 - Terrain-following
- Conditions of input

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

First, we'll talk about techniques for travel (viewpoint movement). We will cover only the implementation of the simple movement itself, but there are other implementation issues that must be considered in general.

One such issue is the control of velocity and/or acceleration. There are many methods for doing this, including gesture, speech controls, sliders, etc.

Another issue is that of world rotation. In systems that are only partially spatially surrounding (e.g. a 4-walled CAVE, or a single screen), the user must be able to rotate the world or his view of the world in order to navigate. In fully surrounding systems (e.g. an HMD) this is not necessary. Next, one must consider whether motion should be constrained in any way, for example by maintaining a constant height or by following the terrain. Finally, at the lowest-level the conditions of input must be considered – that is, when and how does motion begin and end (click to start/stop, press to start, release to stop, stop automatically at target location, etc.)?

Common travel techniques

- Gaze-directed steering
- Pointing
- Map-based travel
- “Grabbing the air”

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

We'll discuss four common techniques.

Gaze-directed steering technique

- Move viewpoint in direction of “gaze”
- Gaze direction determined from head tracker
- Cognitively simple
- Doesn't allow user to look to the side while traveling

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Gaze-directed steering is probably the most common 3D travel technique, although the term “gaze” is really misleading. Usually no eye tracking is being performed, so the direction of gaze is inferred from the head tracker orientation. This is a simple technique, both to implement and to use, but it is somewhat limited in that you cannot look around while moving.

See: Mine, M. (1995). *Virtual Environment Interaction Techniques* (Technical Report TR95-018): UNC Chapel Hill CS Dept.

Gaze-directed steering implementation

- Each frame while moving:
 - Get head tracker information
 - Transform vector $[0,0,-1]$ in head CS to $v=[x,y,z]$ in world CS
 - Normalize v: $\hat{v} = \frac{v}{\|v\|}$
 - Translate viewpoint by $(\hat{v}_x, \hat{v}_y, \hat{v}_z) \times \text{current_velocity}$

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

To implement gaze-directed steering, you set up a callback function that executes before each frame is rendered. Within this callback, you first obtain the head tracker information (usually in the form of a 4x4 matrix). This matrix gives you a transformation between the base tracker CS and the head tracker CS. By also considering the transformation between the world CS and the base tracker CS (if any), you can get the total composite transformation. Now you consider the vector $[0,0,-1]$ in head tracker space (the negative z-axis, which usually points out the front of the tracker). This vector, expressed in world coordinates, is the direction you want to move. Now all that's left to do is to normalize this vector, multiply it by the speed, and then translate the viewpoint by this amount in world coordinates.

Note: current "velocity" is in units/frame. If you want true velocity (units/second), you must keep track of the time between frames and then translate the viewpoint by an amount proportional to that time.

Pointing technique

- Also a steering technique
- Use hand tracker instead of head tracker
- Slightly more complex, cognitively
- Allows travel and gaze in different directions – good for relative motion

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Pointing is also a steering technique (where the user continuously specifies the direction of motion). In this case, the hand's orientation is used to determine direction. This technique is somewhat harder to learn for some users, but is more flexible than gaze-directed steering.

See: Mine, M. (1995). *Virtual Environment Interaction Techniques* (Technical Report TR95-018): UNC Chapel Hill CS Dept., and
Bowman, D. A., Koller, D., & Hodges, L. F. (1997). *Travel in Immersive Virtual Environments: an Evaluation of Viewpoint Motion Control Techniques*. Proceedings of the Virtual Reality Annual International Symposium, 45-52.

Pointing implementation

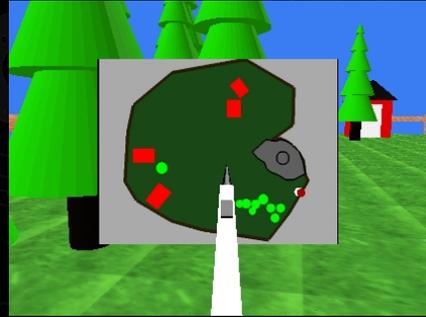
- Each frame while moving:
 - Get hand tracker information
 - Transform vector $[0,0,-1]$ in hand CS to $v=[x,y,z]$ in world CS
 - Normalize v: $\hat{v} = \frac{v}{\|v\|}$
 - Translate viewpoint by $(\hat{v}_x, \hat{v}_y, \hat{v}_z) \times \text{current_velocity}$

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Pointing is implemented in exactly the same way as gaze-directed steering, except that the hand tracker is used instead of the head tracker.

Map-based travel technique

- User represented by icon on 2D map
- Drag icon with stylus to new location on map
- When released, viewpoint animated smoothly to new location



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The map-based technique is a target-based travel technique. The user is represented as an icon on a 2D map of the environment. To travel, the user drags this icon to a new position on the map. When the icon is dropped, the system smoothly animates the user from the current location to the new location indicated by the icon.

See: Bowman, D., Wineman, J., Hodges, L., & Allison, D. (1998). Designing Animal Habitats Within an Immersive VE. *IEEE Computer Graphics & Applications*, 18(5), 9-13.

Map-based travel implementation

- **Must know**
 - Map scale relative to world: s
 - Location of world origin in map CS: $o=(x_o, y_o, z_o)$
- **On button press:**
 - If stylus intersects user icon, then each frame:
 - Get stylus position in map CS: (x, y, z)
 - Move icon to $(x, 0, z)$ in map CS

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

To implement this technique, you need to know two things about the way the map relates to the world. First, you need to know the scale factor. Second, you need to know which point on the map represents the origin of the world CS. We assume here that the map model is originally aligned with the world (i.e. the x direction on the map, in its local CS, represents the x direction in the world CS).

When the user presses the button and is intersecting the user icon on the map, then you need to move the icon with the stylus each frame. You cannot simply attach the icon to the stylus, because you want the icon to remain on the map even if the stylus does not. To do this, you must first find the position of the stylus **in the map CS**. This may require a transformation between coordinate systems, since the stylus is not a child of the map. The x and z coordinates of the stylus position are the point to which the icon should be moved.

We do not cover here what happens if the stylus is dragged off the map, but the user icon should “stick” to the side of the map until the stylus is moved back inside the map boundaries, since we don’t want the user to move outside the world.

Map-based travel implementation (cont.)

- **On button release:**
 - Get stylus position in map CS: (x, y, z)
 - Move icon to $(x, 0, z)$ in map CS
 - Desired viewpoint: $p_v = (x_v, y_v, z_v)$ where
 - $x_v = (x - x_o)/s$
 - $z_v = (z - z_o)/s$
 - $y_v = \text{desired height at } (x_v, y_v)$
 - Move vector: $m = (x_v - x_{curr}, y_v - y_{curr}, z_v - z_{curr}) * (\text{velocity}/\text{distance})$
 - Each frame for $(\text{distance}/\text{velocity})$ frames: translate viewpoint by m

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

When the button is released, the icon is set to its final position, using the same method as before.

Now we need to calculate the desired position of the viewpoint in the world. This position is calculated using a transformation from the map CS to the world CS, which is detailed for you here. First, you must find the offset in the map CS from the point corresponding to the world origin. Then, you divide by the map scale (if the map is 1/100 the size of the world, this corresponds to multiplying by 100). This gives us the x and z coordinates of the desired viewpoint position. Since the map is 2D, we can't get a y coordinate from it. Therefore, the technique should have some way of calculating the desired height at the new viewpoint. In the simplest case, this might be constant. In other cases, it might be based on the terrain height at that location or some other factors.

Now that we know the desired viewpoint, we have to set up the animation of the viewpoint. The move vector m represents the amount of translation to do each frame (we are assuming a linear path). To find m , we subtract the desired position from the current position (the total movement required), divide this by the distance between the two points (calculated using the distance formula), and multiply by the desired velocity, so that m gives us the amount to move in each dimension each frame. The only remaining calculation is the number of frames this movement will take: distance/velocity frames. Note that again velocity is measured here in units/frame, not units/second, for simplicity.

Grabbing the air technique

- Use hand gestures to move yourself through the world
- Metaphor of pulling a rope
- Often a 2-handed technique
- May be implemented using Pinch Gloves™

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The “grabbing the air” technique uses the metaphor of literally grabbing the world around you (usually empty space), and pulling yourself through it using hand gestures. This is similar to pulling yourself along a rope, except that the “rope” exists everywhere, and can take you in any direction.

This technique may be done with one or two hands, and is often implemented using Pinch Gloves™.

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

Grabbing the air implementation (one-handed)

- **On pinch:**
 - Obtain initial hand position in world CS: $(x_{1w} \ y_{1w} \ z_{1w})$
- **Each frame until release:**
 - Obtain current hand position in world CS: $(x'_{1w} \ y'_{1w} \ z'_{1w})$
 - Hand motion vector: $m = ((x'_{1w} \ y'_{1w} \ z'_{1w}) - (x_{1w} \ y_{1w} \ z_{1w}))$
 - Translate world by m (or viewpoint by $-m$)
 - $(x_{1w} \ y_{1w} \ z_{1w}) = (x'_{1w} \ y'_{1w} \ z'_{1w})$
- **Cannot simply attach objects to hand - do not want to match hand rotations**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

We'll describe a simple one-handed technique. This could be extended to two hands by running this algorithm for each hand separately, making sure that only one hand is activated at a time.

When the initial pinch or button press is detected, simply obtain the position of the hand in the world CS.

Then, every frame until the pinch is released, get a new hand position, subtract it from the old one, and move the objects in the world by this amount. Alternately, you can leave the world fixed, and translate the viewpoint by the opposite vector. Before exiting the callback, be sure to update the "old" hand position for use on the next frame.

It is tempting to implement this technique simply by attaching the world to the hand, but this will have the undesirable effect of also rotating the world when the hand rotates, which can be quite disorienting.

You can do simple constrained motion simply by ignoring one or more of the components of the hand position (e.g. only consider x and z to move at a constant height).

Implementation issues for selection techniques

- How to indicate selection event
- Object intersections
- Feedback
 - Graphical
 - Aural
 - Tactile
- Virtual hand avatar
- List of selectable objects

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Next, we'll discuss object selection. We must first note that selection and manipulation are intimately related, and that several of the techniques described here can also be used for manipulation.

There are several common issues for the implementation of selection techniques. One of the most basic is how to indicate that the selection event should take place (e.g. you are touching the desired object, now you want to pick it up). This is usually done via a button press, gesture, or voice command, but it might also be done automatically if the system can infer the user's intent. You also have to have efficient algorithms for object intersections for many of these techniques. We'll discuss a couple of possibilities. The feedback you give to the user regarding which object is about to be selected is also very important. Many of the techniques require an avatar (virtual representation) for the user's hand. Finally, you should consider keeping a list of objects that are "selectable", so that your techniques do not have to test every object in the world for selection, increasing efficiency.

Common selection techniques

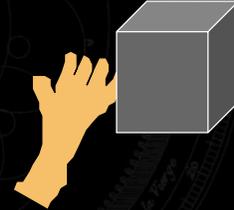
- Simple virtual hand
- Ray-casting
- Sticky finger (occlusion)
- Go-go (arm-extension)

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

We'll discuss four selection techniques.

Simple virtual hand technique

- One-to-one mapping between physical and virtual hands
- Object can be selected by “touching” or intersecting v. hand with object



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The most common technique is the simple virtual hand, which does “real-world” selection via direct “touching” of virtual objects. In the absence of haptic feedback, this is done by intersecting the virtual hand (which is at the same location as the physical hand) with a virtual object.

Implementing this technique is simple, provided you have a good intersection/collision algorithm. Often, intersections are only performed with axis-aligned bounding boxes or bounding spheres rather than with the actual geometry of the objects.

Ray-casting technique

- “Laser pointer” attached to v. hand
- First object intersected by ray may be selected
- User only needs to control 2 DOFs
- Empirically proven to perform well



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Another common technique is ray-casting. This technique uses the metaphor of a laser pointer – an infinite ray extending from the virtual hand. The first object intersected along the ray is eligible for selection. This technique is efficient, based on experimental results, and only requires the user to vary 2 degrees of freedom (pitch and yaw of the wrist) rather than the 3 DOFs required by the simple virtual hand and other location-based techniques.

See: Mine, M. (1995). *Virtual Environment Interaction Techniques* (Technical Report TR95-018): UNC Chapel Hill CS Dept.

Ray-casting implementation

- **Naïve: intersect ray with each polygon**
 - Parametric equation
 - Only consider intersections with $t > 0$
- **Better: transform vertices (or bounding box) to hand's CS**
 - Drop new z coordinate of every vertex
 - Ray intersects polygon iff $(0,0)$ is in the polygon
 - Count the number of times the polygon edges cross the positive x-axis



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

There are many ways to implement ray-casting.

A brute-force approach would calculate the parametric equation of the ray, based on the hand's position and orientation. First, as in the pointing technique for travel, find the world CS equivalent of the vector $[0,0,-1]$. This is the direction of the ray. If the hand's position is represented by (x_h, y_h, z_h) , and the direction vector is (x_d, y_d, z_d) , then the parametric equations are given by:

$$x(t) = x_h + x_d t$$

$$y(t) = y_h + y_d t$$

$$z(t) = z_h + z_d t$$

Only intersections with $t > 0$ should be considered, since we don't want to count intersections "behind" the hand. If you use this method and you want to determine whether the actual geometry has been intersected, you should first test the intersection with the bounding box so that many cases can be trivially rejected.

-continued on the next page

Another method might be more efficient. In this method, instead of looking at the hand orientation in the world CS, we consider the selectable objects to be in the hand's CS, by transforming their vertices or their bounding boxes. This might seem quite inefficient, because there is only one hand, while there are many polygons in the world. However, we assume we have limited the objects by using a selectable objects list, and the intersection test we will describe is much more efficient.

Once we have transformed the vertices or bounding boxes, we drop the z coordinate of each vertex. This maps the 3D polygon onto a 2D plane (the xy plane in the hand CS). Since the ray is $[0, 0, -1]$ in this CS, we can see that in this 2D plane, the ray will intersect the polygon if and only if the point $(0, 0)$ is in the polygon (see the figure). We can easily determine this with an algorithm that counts the number of times the edges of the 2D polygon cross the positive x-axis. If there are an odd number of crossings, the origin is inside, if even, the origin is outside.

Not discussed on this slide is a third method of implementation. The OpenGL graphics package includes a "selection" concept. By rendering the selectable objects to an offscreen buffer from the point of view of the hand, you can determine which objects are intersected by the ray, and which one is closest.

Occlusion technique

- Image-plane technique
- truly 2D
- Occlude/cover desired object with selector object (e.g. finger)
- Nearest object along ray from eye through finger may be selected



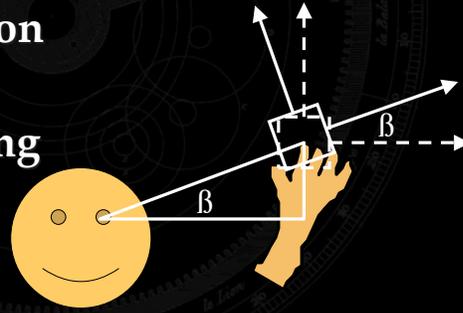
SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Next, we'll cover the occlusion technique (also called the “sticky finger” technique). This technique works in the plane of the image – that is, you select an object by “covering” it with the virtual hand so that it is occluded from your point of view. Geometrically, this means that a ray is emanating from your eye, going through your finger, and then intersecting an object.

See: Pierce, J., Forsberg, A., Conway, M., Hong, S., Zeleznik, R., & Mine, M. (1997). *Image Plane Interaction Techniques in 3D Immersive Environments*. Proceedings of the ACM Symposium on Interactive 3D Graphics, 39-44.

Occlusion implementation

- Special case of ray-casting technique
- Must consider position of eye/camera
- Can use 2nd ray-casting algorithm; requires special object



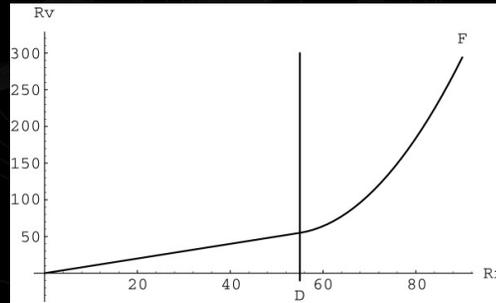
SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

This technique can be implemented in the same ways as the ray-casting technique, since it is also using a ray. If you are doing the brute-force ray intersection algorithm, you can simply define the ray's direction by subtracting the finger position from the eye position.

However, if you are using the 2nd algorithm, you require an object to define the ray's coordinate system. This can be done in two steps. First, create an empty object, and place it at the hand position, aligned with the world CS (the dotted lines in the figure). Next, determine how to rotate this object/CS so that it is aligned with the ray direction. In the figure, a 2D example is given. The angle can be determined using the positions of the eye and hand, and some simple trigonometry. In 3D, two rotations must be done in general to align the new object's CS with the ray.

Go-Go technique

- Arm-extension technique
- Like simple v. hand, touch objects to select them
- Non-linear mapping between physical and virtual hand position
- Local and distant regions



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The Go-Go technique is based on the simple virtual hand, but it introduces a non-one-to-one mapping between the physical hand and the virtual hand, so that the user's reach is greatly extended. This is called an arm-extension technique.

The graph shows the mapping between the physical hand distance from the body on the x-axis and the virtual hand distance from the body on the y-axis. There are two regions. When the physical hand is at a depth less than a threshold 'D', the one-to-one mapping applies. Outside D, a non-linear mapping is applied, so that the farther the user stretches, the faster the virtual hand moves away.

See: Poupyrev, I., Billinghamurst, M., Weghorst, S., & Ichikawa, T. (1996). *The Go-Go Interaction Technique: Non-linear Mapping for Direct Manipulation in VR*. Proceedings of the ACM Symposium on User Interface Software and Technology, 79-80.

Go-Go implementation

- Requires “torso position” t - tracked or inferred
- Each frame:
 - Get physical hand position h in world CS
 - Calculate physical distance from torso $d_p = \text{dist}(h, t)$
 - Calculate virtual hand distance $d_v = \text{gogo}(d_p)$
 - Normalize torso-hand vector $th = \frac{h-t}{\|h-t\|}$
 - V. hand position $v = t + d_v * th$ (in world CS)

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

To implement Go-Go, we first need the concept of the position of the user's body. This is needed because we stretch our hands out from the center of our body, not from our head (which is usually the position that is tracked). I have implemented this using an inferred torso position, which is defined as a constant offset in the negative y direction from the head. You could also place a tracker on the user's torso.

Before rendering each frame, you get the physical hand position in the world CS, and then calculate its distance from the torso object using the distance formula. The virtual hand distance can then be obtained by applying the function shown in the graph on the previous slide. I have used the function $d^{2.3}$ (starting at the point (D,D)) as a useful function in my environments, but the exponent used depends on the size of the environment and the desired accuracy of selection at a distance.

Now that we know the distance at which to place the virtual hand, we need to determine its position. The most common implementation is to keep the virtual hand on the ray extending from the torso and going through the physical hand. Therefore, if we get a vector between these two points, normalize it, multiply it by the distance, then add this vector to the torso point, we obtain the position of the virtual hand.

Implementation issues for manipulation techniques

- Integration with selection technique
- Disable selection and selection feedback while manipulating
- What happens upon release?

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Next we turn to techniques for 3D object manipulation. As we noted earlier, manipulation is connected with selection, because an object must be selected before you can manipulate it. Thus, one important issue for any manipulation technique is how well it integrates with the chosen selection technique. Many techniques, as we have said, do both: e.g. simple virtual hand, ray-casting, and go-go. Another issue is that when an object is being manipulated, you should take care to disable the selection technique and the feedback you give the user for selection. If this is not done, then serious problems can occur if, for example, the user tries to release the currently selected object but the system also interprets this as trying to select a new object. Finally, you should think in general about what happens when the object is released. Does it remain at its last position, possibly floating in space? Does it snap to a grid? Does it fall via gravity until it contacts something solid? The application requirements will determine this choice.

Common manipulation techniques

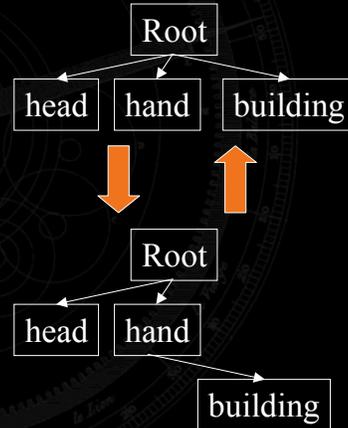
- Simple virtual hand
- HOMER
- Scaled-world grab
- World-in-miniature

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

We'll discuss four 3D object manipulation techniques.

Simple virtual hand technique

- Attach object to virtual hand, by making object a child of the hand (w/out moving object)
- On release, reattach object to world (w/out moving object)
- Also applies to Go-Go (and other arm-extension techniques) and ray-casting

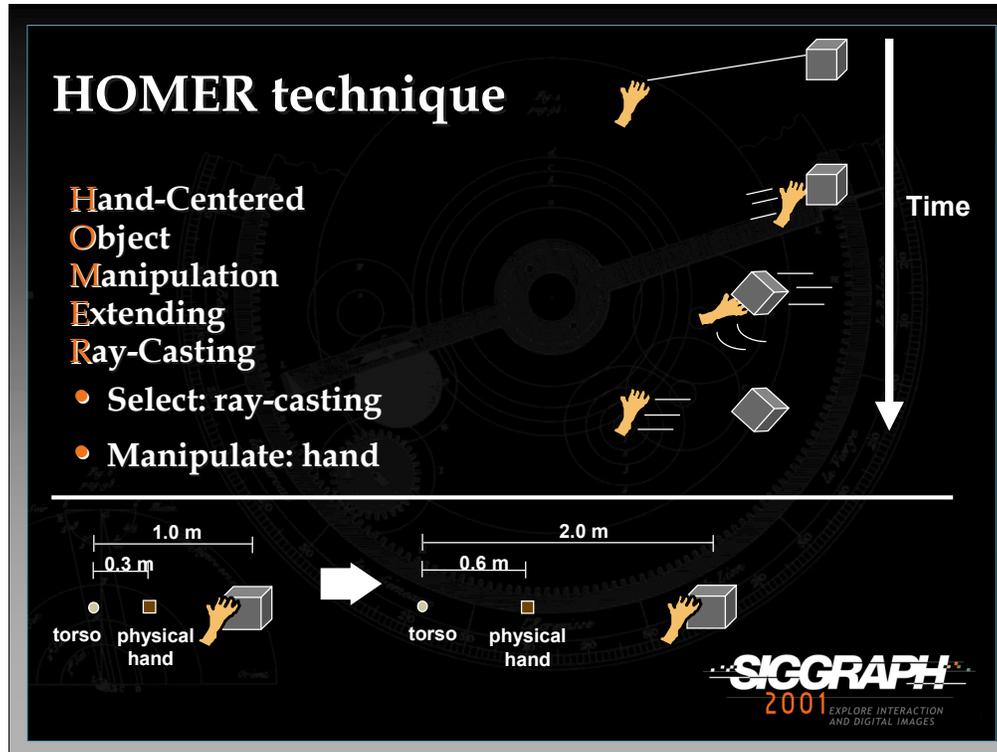


SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

We already saw the simple virtual hand technique for selection. When this technique is used for object manipulation, the implementation is quite easy. It simply involves making a change to the scene graph by attaching the selected object to the virtual hand. Then, as the virtual hand moves and rotates, the selected object will inherit those transformations. When the object is released, it should just be reattached to its earlier location in the tree.

The only tricky issue here is that you must ensure when grabbing or releasing the object that it does not move (in the world CS). If you simply make the object a child of the hand, it may move since its position is now being interpreted relative to a new CS (the hand's). To be completely general, then, you must get the object's position p in the world CS first, then do the attachment, then calculate p 's location in the hand CS, then move the object to that position (relative to the hand). The opposite transformation is done upon release.

This same basic procedure works for other techniques that simply attach the object to the selector, like Go-Go and ray-casting.



The HOMER technique uses ray-casting for selection and then moves the virtual hand to the object for hand-centered manipulation. The depth of the object is based on a linear mapping, as shown in the figure at the bottom. The initial torso-physical hand distance is mapped onto the initial torso-object distance, so that moving the physical hand twice as far away also moves the object twice as far away. Also, moving the physical hand all the way back to the torso moves the object all the way to the user's torso as well.

See: Bowman, D., & Hodges, L. (1997). *An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments*. Proceedings of the ACM Symposium on Interactive 3D Graphics, 35-38.

HOMER implementation

- Requires torso position t
- Upon selection, detach virtual hand from tracker, move v. hand to object position in world CS, and attach object to v. hand (w/out moving object)
- Get physical hand position h and distance $d_h = \text{dist}(h, t)$
- Get object position o and distance $d_o = \text{dist}(o, t)$

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Like Go-Go, HOMER requires a torso position, because you want to keep the virtual hand on the ray between the user's body (torso) and the physical hand. The problem here is that HOMER moves the virtual hand from the physical hand position to the object upon selection, and it is not guaranteed that the torso, physical hand, and object will all line up at this time. Therefore, in my implementation, I calculate where the virtual hand would be if it were on this ray initially, then calculate the offset to the position of the virtual object, and maintain this offset throughout manipulation.

When an object is selected via ray-casting, you must first detach the virtual hand from the hand tracker. This is due to the fact that if it remained attached but you move the virtual hand model away from the physical hand location, a rotation of the physical hand will cause a rotation and translation of the virtual hand. You then move the virtual hand in the world CS to the position of the selected object, and attach the object to the virtual hand in the scene graph (again, without moving the object in the world CS).

To implement the linear depth mapping, we need to know the initial distance between the torso and the physical hand, and between the torso and the selected object. The ratio d_o/d_h will be the scaling factor.

HOMER implementation (cont.)

- **Each frame:**

- Copy hand tracker matrix to v. hand matrix (to set orientation)

- Get physical hand position h_{curr} and distance:

$$d_{h-curr} = dist(h_{curr}, t)$$

- V. hand distance $d_{vh} = d_{h-curr} \times \left(\frac{d_o}{d_h} \right)$

- Normalize torso-hand vector $th_{curr} = \frac{h_{curr} - t}{\|h_{curr} - t\|}$

- V. hand position $vh = t + d_{vh} * (th_{curr})$



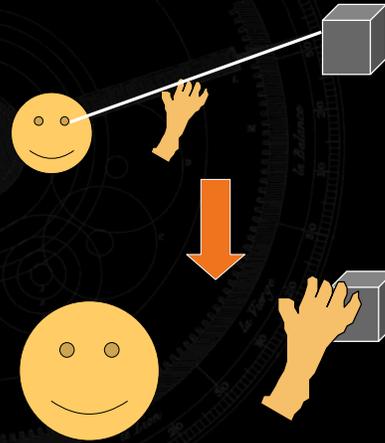
Now, each frame you need to set the position and orientation of the virtual hand. The selected object is attached to the virtual hand, so it will follow along.

Setting the orientation is relatively easy. You can simply copy the transformation matrix for the hand tracker to the virtual hand, so that their orientation matches.

To set the position, we need to know the correct depth and the correct direction. The depth is found by applying the linear mapping to the current physical hand depth. The physical hand distance is simply the distance between it and the torso, and we multiply this by the scale factor d_o/d_h to get the virtual hand distance. We then obtain a normalized vector between the physical hand and the torso, multiply this vector by the v. hand distance, and add the result to the torso position to obtain the virtual hand position.

Scaled-world grab technique

- Often used w/ occlusion
- At selection, scale user up (or world down) so that v. hand is actually touching selected object
- User doesn't notice a change in the image until he moves



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The scaled-world grab technique is often used with occlusion selection. The idea is that since you are selecting the object in the image plane, you can use the ambiguity of that single image to do some magic. When the selection is made, the user is scaled up (or the world is scaled down) so that the virtual hand is actually touching the object that it was occluding. If the user doesn't move (and the graphics are not stereo), there is no perceptual difference between the images before and after the scaling. However, when the user starts to move the object and/or his head, he realizes that he is now a giant (or that the world is tiny) and he can manipulate the object directly, just like the simple virtual hand.

See: Mine, M., Brooks, F., & Sequin, C. (1997). *Moving Objects in Space: Exploiting Proprioception in Virtual Environment Interaction*. Proceedings of ACM SIGGRAPH, 19-26, and

Pierce, J., Forsberg, A., Conway, M., Hong, S., Zeleznik, R., & Mine, M. (1997). *Image Plane Interaction Techniques in 3D Immersive Environments*. Proceedings of the ACM Symposium on Interactive 3D Graphics, 39-44.

Scaled-world grab implementation

- **At selection:**
 - Get world CS distance from eye to hand d_{eh}
 - Get world CS distance from eye to object d_{eo}
 - Scale user (entire user subtree) uniformly by d_{eo}/d_{eh}
 - Ensure that eye remains in same position
 - Attach selected object to v. hand (w/out moving object)
- **At release:**
 - Re-attach object to world (w/out moving object)
 - Scale user uniformly by d_{eh}/d_{eo}
 - Ensure that eye remains in same position

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

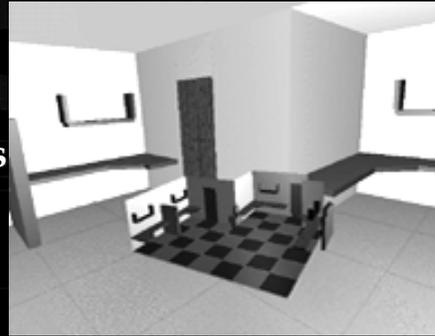
To implement scaled-world grab, you must do the correct actions at the time of selection and release. Nothing special needs to be done in between, because the object is simply attached to the virtual hand, as in the simple virtual hand technique.

At the time of selection, you want to scale the user by the ratio (distance from eye to object / distance from eye to hand). This scaling needs to take place with the eye as the fixed point, so that the eye does not move, and should be uniform in all three dimensions. Finally, you attach the virtual object to the virtual hand as you would normally.

At the time of release, the opposite actions are done in reverse. You re-attach the object to the world, and scale the user uniformly by the reciprocal of the scaling factor, again using the eye as a fixed point.

World-in-miniature (WIM) technique

- “Dollhouse” world held in user’s hand
- Miniature objects can be manipulated directly
- Moving miniature objects affects full-scale objects
- Can also be used for navigation



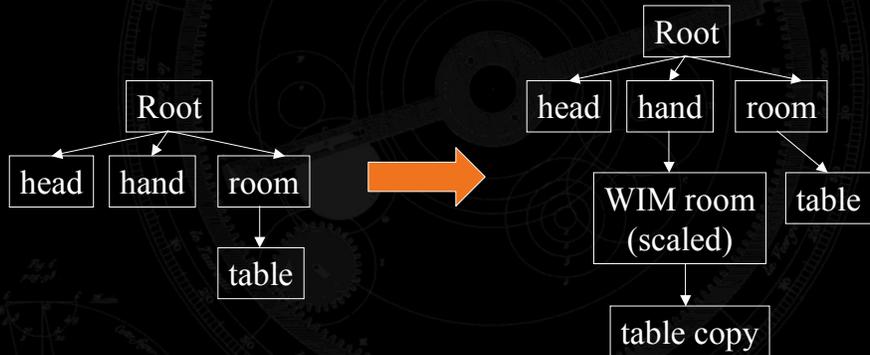
SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The world-in-miniature (WIM) technique uses a small “dollhouse” version of the world to allow the user to do indirect manipulation of the objects in the environment. Each of the objects in the WIM is selectable using the simple virtual hand technique, and moving these objects causes the full-scale objects in the world to move in a corresponding way. The WIM can also be used for navigation by including a representation of the user, in a way similar to the map-based travel technique, but including the 3rd dimension.

See: Stoakley, R., Conway, M., & Pausch, R. (1995). *Virtual Reality on a WIM: Interactive Worlds in Miniature*. Proceedings of CHI: Human Factors in Computing Systems, 265-272, and

Pausch, R., Burnette, T., Brockway, D., & Weiblen, M. (1995). *Navigation and Locomotion in Virtual Worlds via Flight into Hand-Held Miniatures*. Proceedings of ACM SIGGRAPH, 399-400.

WIM implementation



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

To implement the WIM technique, you first need to create the WIM. Consider this example, where you have a room with a table object in it. The WIM is represented as a scaled down version of the room, and it attached to the virtual hand. The table object does not need to be scaled, because it will inherit the scaling from its parent (the WIM room). Thus, the table object can simply be copied within the scene graph.

WIM implementation (cont.)

- **On selection:**
 - Determine which full-scale object corresponds to the selected miniature object
 - Attach miniature object to v. hand (w/out moving object)
- **Each frame:**
 - Copy local position matrix of miniature object to corresponding full-scale object

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

When an object in the WIM is selected using the simple virtual hand technique, you first have to match this object to the corresponding full-scale object. Keeping a list of pointers to these objects is an efficient way to do this step. The miniature object is attached to the virtual hand, just as in the simple technique.

While the miniature object is being manipulated, you can simply copy its position matrix (in its local CS, relative to its parent – the WIM) to the position matrix of the full-scale object. Since we want the full-scale object to have the same position in the full-scale world CS as the miniature object does in the scaled-down WIM CS, this is all that is necessary to move the full-scale object correctly.

Common system control techniques

- Virtual menus
- Tool selectors (belts, palettes, chests)
- Speech commands
- Pen & tablet technique

- For the most part, these only require a selection technique
- Good visual feedback is necessary

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

System control is a wide-ranging topic, and there are many different techniques, some of which are listed here. For the most part, these techniques are not difficult to implement, since they mostly involve selection, which we've already covered. For example, virtual menu items might be selected using ray-casting. For all of the techniques, good visual feedback is required, since the user needs to know not only what he is selecting, but what will happen when he selects it.

Pen & tablet technique



I only want to touch on one system control technique, because of its widespread use. The pen & tablet technique uses a physical pen and tablet (see left image). In the virtual world, the user sees a virtual pen and tablet, and a 2D interface on the surface of the virtual tablet (right image). The physical devices provide near-field haptics and constraints that make such an interface easy to use.

See: Angus, I., & Sowizral, H. (1995). *Embedding the 2D Interaction Metaphor in a Real 3D Virtual Environment*. Proceedings of SPIE, Stereoscopic Displays and Virtual Reality Systems, 282-293, and

Schmalsteig, D., Encarnacao, L., & Szalzvári, Z. (1999). *Using Transparent Props For Interaction with The Virtual Table*. Proceedings of the ACM Symposium on Interactive 3D Graphics, 147-154.

Pen & tablet implementation

- **Registration of physical and virtual is crucial**
 - Physical and virtual tablet same size
 - Origin of virtual tablet at location tracker is attached
 - Still may need controls to tweak positions of avatars
- **Useful to have system report on the offset of the stylus tip from the origin in tablet CS**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

For the implementation of this technique, the most crucial thing is the registration (correspondence) between the physical and virtual pens and tablets. The tablets, especially, must be the same size and shape so that the edge of the physical tablet, which the user can feel, corresponds to the edge of the virtual tablet as well. In order to make tracking easy, the origin of the tablet model should be located at the point where the tracker is attached to the physical tablet, so that rotations work properly. Even with care, it's difficult to do these things exactly right, so a final tip is to include controls within the program to tweak the positions and/or orientations of the virtual pen and tablet, so that you can bring them into registration if there's a problem.

Another useful function to implement is the ability for the system to report (for example when a callback function is called) the position of the stylus tip in the tablet CS, rather than in the world or user CSs. This can be used for things like the map-based travel technique described earlier.

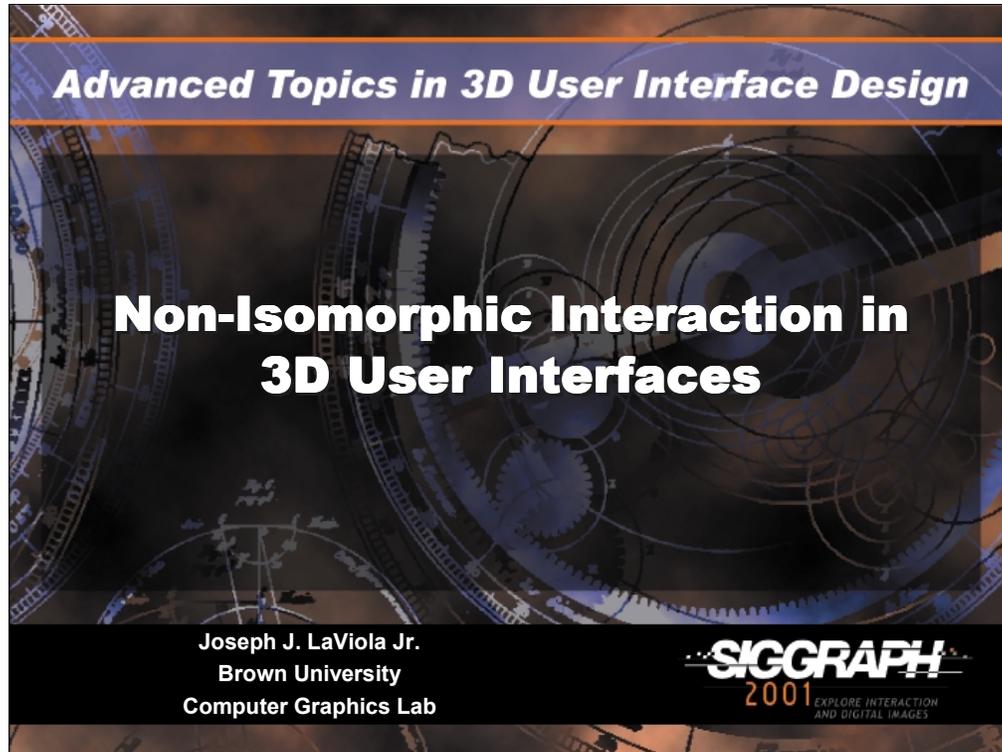
Conclusions

- **Implementation details are crucial for usability**
- **Ease of coding \neq ease of use**
- **Implementation of interaction techniques should be taken into consideration when designing 3D development toolkits**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

In this lecture, we've given an overview of the implementation of several important 3D interaction techniques. There are still more details that could not be covered here, but hopefully this gives the developer the basic tools needed to implement many types of 3D interfaces.

This information is also important because “the devil is in the details.” A poorly implemented technique will likely have usability problems. For example, if the user is not scaled with the eye as a fixed point in the scaled-world grab technique, selection will cause the user to “jump”, which could disorient users or even make them sick. Another important point is that the easiest techniques to implement are not necessarily the easiest to use, and vice-versa. It may take a lot of work to develop a usable technique. Finally, we hope that future tools for the development of 3D applications will include support for the implementation of interaction techniques, both through the inclusion of standard techniques and technique components, and through functionality that matches the needs of interaction designers.



Non-Isomorphic Interaction in 3D User Interfaces

Joseph J LaViola Jr

Ph.D Candidate

Brown University,

Department of Computer Science

Providence, Rhode Island

Lead Consultant and Founder

JJL Interface Consultants, Inc.

http: <http://www.cs.brown.edu/people/jjl>

email: jjl@cs.brown.edu

Goals and Motivation

- **Describe non-isomorphic interaction techniques**
 - spatial rotation of virtual objects
 - virtual environment navigation
- **Present mathematical foundations**
- **Discuss importance of non-isomorphic ideas**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

In this lecture, we are going to discuss non-isomorphic interaction techniques and present specific examples for navigation and object rotation in 3D user interfaces. We will also present some mathematical foundations for developing these techniques and examine their utility in 3D applications.

Lecture Outline

- **Isomorphic vs. non-isomorphic approaches**
- **Non-Isomorphic 3D Spatial Rotation**
 - rotational space and quaternions
 - linear and non-linear approaches
 - absolute vs. relative mappings
- **Amplified Non-Linear Rotation for VE navigation**
- **Non-Linear translation for VE navigation**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

After describing the differences between isomorphic and non-isomorphic interaction philosophies, we will present a framework for developing non-isomorphic object rotation techniques, discuss design trade-offs, and present some experimental evaluations of a specific technique. Next, we describe a specific technique for amplifying a user's rotation in a surround screen virtual environment (SSVE). Finally, we present a technique for moving through a virtual environment using a leaning metaphor and non-linear translations.

Isomorphic vs. Non-Isomorphic Philosophies

- **Human-Machine interaction**
 - input device
 - display device
 - transfer function (control to display mapping)
- **Isomorphic – one-to-one mapping**
- **Non-isomorphic – scaled linear/non-linear mapping**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

There are three basic components of any human-machine interface:

- 1) input devices which capture user actions
- 2) display devices which present the effect of these actions back to the user
- 3) transfer functions (control to display mappings) which map device movements into movements of controlled display or interface elements.

These control to display mapping functions (CD mappings) significantly impact user performance and have been an active research area in 3D user interfaces where two competing philosophies have emerged. The isomorphic approach suggests a strict geometric isomorphism (i.e. one-to-one mapping) between physical and virtual objects on the grounds that it is the most natural and therefore the most useful.

Although isomorphism is often more natural, it has number of shortcomings. First, isomorphic mappings are often impractical because of input device constraints such as limited tracking range. Second, isomorphism is often ineffective due to limits of human operators such as anatomical constraints.

In contrast, the non-isomorphic approach suggests that manipulation mappings and techniques can deviate from strict realism, providing users with “magic” virtual tools. Instead of one-to-one isomorphic mappings, non-isomorphic techniques use scaled linear and non-linear mapping functions which, in effect, give users more power to manipulate virtual objects and navigate through 3D worlds.

Non-Isomorphic 3D Spatial Rotation

- **Important advantages**

- manual control constrained by human anatomy
- more effective use of limited tracking range (i.e vision-based tracking)
- additional tools for fine tuning interaction techniques

- **Questions**

- faster?
- more accurate?



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Non-isomorphic mappings are not an entirely new idea; they have been used for decades in a variety of everyday controls such as dials, wheels, and levers. In the context of 3D user interfaces, non-isomorphic techniques have been primarily designed for dealing with translation components in multiple DOF input. For 3D rotations, most techniques only use the simple one-to-one mappings between virtual objects and the 3D input device. What advantages can we gain from using a non-isomorphic approach to 3D spatial rotation? One of the biggest gains that the non-isomorphic approach provides is a method for handling manual control constraints. For example, with an isomorphic approach, rotating an object a full 360 degrees is extremely difficult in one motion since our arms have limited rotational movement about the elbow. A non-isomorphic mapping could eliminate this problem by mapping the user's, clearly limited, interaction space onto the full 360 degree virtual object space. Other advantages of the non-isomorphic approach include the ability to make better use of limited tracking range some input devices have such as a desktop VR camera tracking system. Another question we must consider is how these interaction approaches improve or hinder user performance; a question that we will begin to answer with some usability evaluations described later on in the lecture.

Rotational Space

- **Rotations in 3D space are a little tricky**
 - do not follow laws of Euclidian geometry
- **Space of rotations is not a vector space**
- **Represented as a closed and curved surface**
 - 4D sphere or manifold
- **Quaternions provide a tool for describing this surface**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Rotations in 3D space are more confusing than they appear since they do not obey the laws of Euclidean geometry. For example, rotate an object in a certain direction and it will eventually come back to its initial orientation. The reason for this is that the space of rotations is not a vector space, but a closed and curved surface that can be represented as a 4D sphere. Quaternions provide us with a tool for describing rotations within the context of this 4D sphere and we will use them to develop a basic mathematical framework for designing non-isomorphic transfer functions.

Quaternions

- Four-dimensional vector (v, w) where v is a 3D vector and w is a real number
- A quaternion of unit length can be used to represent a single rotation about a unit axis \hat{u} and angle θ as

$$q = \left(\sin\left(\frac{\theta}{2}\hat{u}\right), \cos\left(\frac{\theta}{2}\right) \right) = e^{\frac{\theta}{2}\hat{u}}$$

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Quaternions provide an efficient mechanism to describe and operate 3D rotations. Quaternion q is a four-dimensional vector often represented as a pair (\mathbf{v}, w) where \mathbf{v} is a 3D vector and w is a real number. Given q , we can compute its length and inverse; given quaternion q' , we can compute their multiplication and dot product. The set of all unit quaternions forms a unit sphere in four dimensions and each point on the surface of the sphere represents an orientation of a rigid body. Euler showed that a combination of any number of rotations can be represented as a single rotation from a reference orientation. If no reference orientation is explicitly specified, a quaternion defines the rotation from the identity quaternion which is an analog to the origin in a vector space.

Linear 0th Order 3D Rotation

- Let q_c be the orientation of the input device and q_d be the displayed orientation then

$$(1) \quad q_c = \left(\sin\left(\frac{\theta_c}{2} \hat{u}_c\right), \cos\left(\frac{\theta_c}{2}\right) \right) = e^{\frac{\theta_c}{2} \hat{u}_c}$$

$$(2) \quad q_d = \left(\sin\left(\frac{k\theta_c}{2} \hat{u}_c\right), \cos\left(\frac{k\theta_c}{2}\right) \right) = e^{\frac{k\theta_c}{2} \hat{u}_c} = q_c^k$$

- Final equations w.r.t. identity or reference orientation q_o are

$$(3) \quad q_d = q_c^k \quad (4) \quad q_d = (q_c q_o^{-1})^k q_o, \quad k = \text{CD gain coefficient}$$

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Equation 1 shows the quaternion, given an axis of rotation and angle, of a given multiple DOF input device. We can amplify the rotation angle by applying a constant coefficient k (the C-D gain) to it and leaving the rotation axis intact (equation 2). Therefore, the basic equation for a 0th order linear CD gain is the power function shown as equation 3. Equation 3 specifies this rotation given the reference rotation is the identity quaternion. Sometimes we want to compute rotations given an explicitly specified reference rotation. This calculation can be done by connecting the reference quaternion and the input device quaternion, amplifying it, and combining it with the reference quaternion as shown in equation 4.

Non-Linear 0th Order 3D Rotation

- Consider

$$(3) \quad q_d = q_c^k \quad (4) \quad q_d = (q_c q_o^{-1})^k q_o$$

- Let k be a non-linear function as in

$$\omega = 2 \arccos(q_c \cdot q_o) \quad \text{or} \quad \omega = 2 \arccos(w)$$

$$k = F(\omega) = \begin{cases} 1 & \text{if } \omega < \omega_o \\ f(\omega) = 1 + c(\omega - \omega_o)^2 & \text{otherwise} \end{cases}$$

where c is a coefficient and ω_o is the threshold angle



If we consider the linear 0th order rotation equations derived in the previous slide (equations 3 and 4) k is simply a constant C-D gain. We can use the same equations to create non-linear C-D mappings by letting k equal a non-linear function. In the example shown in the slide, F is a non-linear function based on the smallest angle between the reference quaternion and the input device's quaternion. If this angle is below a certain threshold the C-D gain has a constant ratio, otherwise the C-D gain grows in a non-linear fashion according to the distance between the reference and input device quaternion. Note that there are many other non-linear functions we could apply here and we will see another example later in the lecture.

Design Considerations

- Absolute mapping - taken on *i-th* cycle of the simulation loop

$$q_{d_i} = q_{c_i}^k$$

- Relative mapping - taken between the *i-th* and *i-1th* cycle of the simulation loop

$$q_{d_i} = (q_{c_i} q_{c_{i-1}}^{-1})^k q_{d_{i-1}}$$

SIGGRAPH
2001 EXPLORE INTERACTION
AND DIGITAL IMAGES

When using non-isomorphic spatial rotation techniques, we can use either an absolute or relative mapping scheme. With absolute mapping, we get the absolute orientation of the device, the absolute angular displacement relative to the initial, zero orientation, and use it on the *i-th* cycle of the simulation loop to compute the amplified, virtual object orientation. With relative mapping, we amplify only relative changes in the device orientation, i.e. we use the device's relative orientation between the *i-th* and *i-1th* cycle in the simulation loop to compute the virtual object rotation. Using one scheme or the other does make a difference. First, given the same rotation path about the device, these two mappings produce different rotation paths of the displayed object. Second, absolute and relative mappings are quite different from a usability point of view.

Absolute Non-Isomorphic Mapping

- Generally do not preserve directional compliance
- Strictly preserve nulling compliance

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Absolute non-isomorphic mappings generally do not preserve directional compliance. What we mean by this statement is that virtual objects do not always rotate in the same direction as the device. On the other hand, absolute non-isomorphic mappings strictly preserve the nulling compliance, meaning that the input device always returns the virtual object to its initial orientation preserving a consistent correspondence between the origins of the coordinate systems in the physical and virtual spaces. How does this knowledge affect interface design? Absolute mappings have limited utility since users cannot consistently predict the response of the virtual object on the device rotations. However, these mappings can be useful when device rotations do not change the axis much as with viewpoint control using head rotations.

Relative Non-Isomorphic Mapping

- Always maintain directional compliance
- Do not generally preserve nulling compliance

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

In contrast to absolute mappings, relative non-isomorphic mappings always maintain directional compliance and do not generally preserve nulling compliance. This means that relative non-isomorphic mappings can be very efficient in manual control tasks assuming the multiple DOF input device has the right form factor. The right form factor in this case is the device can be freely rotated in the fingers.

Experimental Usability Study

- **Comparison of relative non-isomorphic rotation technique with conventional technique**
- **Hypothesis**
 - rotation tasks will be faster with non-isomorphic approach for large rotations
 - moderate amplified rotations will decrease accuracy
- **Results**
 - subjects performed 13% faster with non-isomorphic approach with no accuracy degradation

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

How effective are non-isomorphic rotation techniques in terms of speed and accuracy? Obviously, a complete series of usability evaluations would be required to determine the effectiveness of these techniques which is beyond the scope of this lecture. However, as an example, we can describe one such evaluation which compares a linear non-isomorphic rotation technique with a relative mapping versus a conventional one-to-one mapping scheme. There are two main hypotheses for this study. First, a relative amplification of multiple DOF input device rotations will allow users to perform a rotation task faster than a traditional approach assuming large range rotations are required. Second, non-isomorphic rotation techniques with moderate amplification will decrease rotational accuracy. Subjects in the study had to rotate a house model from a randomly generated initial orientation to a target orientation using both the isomorphic and non-isomorphic approaches. Results indicated that subject performed the rotation task 13% faster with the non-isomorphic approach with no accuracy degradation. These results show that in this instance, the non-isomorphic approach is the better one.

For more details on this experiment and on what we have discussed so far, refer to Poupyrev's paper, "Non-Isomorphic 3D Rotational Techniques", included in papers section of the course notes.

Amplified Non-Linear Rotation for VE Navigation (1)

- **Users expect the virtual world to exist in any direction**
 - 3-walled Cave does not allow this
 - adapt expected UI to work in restricted environment
- **Amplified rotation allows users to see a full 360 degrees in a 3-walled display**
- **A number of approaches were tested**
 - important to take cybersickness into account

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

At this point in the lecture, we will describe two non-linear non-isomorphic interaction techniques for navigating through virtual environments. The first technique is an amplified non-linear rotation technique which lets user see a full 360 degrees in a semi-immersive display. In a fully immersive VE (such as an HMD), there is generally no need to provide any explicit control for rotating the virtual environment relative to the user, since the user can turn to face any direction. However, when dealing with a semi-immersive display such as a 3-walled Cave, the user only has approximately 270 degree view of the world and requires explicit controls to see what is behind him/her. A non-isomorphic approach can help to alleviate this problem and give the user the ability to see a full 360 degrees in a 3-walled display. A number of linear C-D mappings were initially tested with negative results. This lead to a subtle, non-linear scheme which provided smooth, less disturbing rotations.

Amplified Non-Linear Rotation for VE Navigation (2)

- Apply a non-linear mapping function to the user's waist orientation θ and his or her distance d from the back of the Cave
- Calculate the rotation factor using a scaled 2D Gaussian function

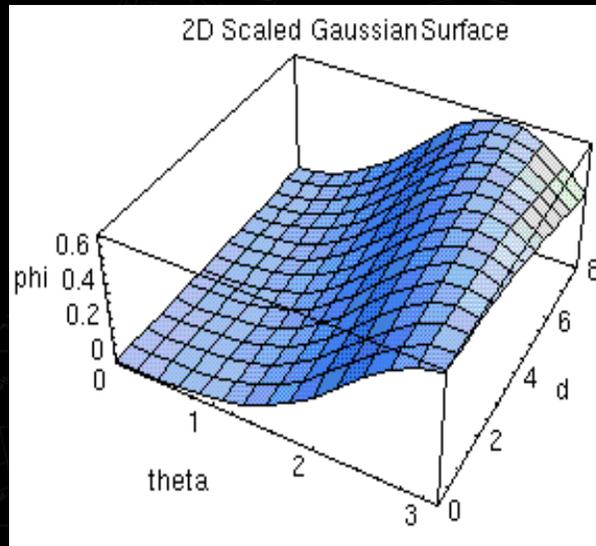
$$\phi = f(\theta, d) = \frac{1}{\sqrt{2\pi\sigma_1}} e^{-\frac{(|\theta| - \pi(1-d/L))^2}{2\sigma_2^2}}$$

- The new viewing angle is $\theta_{new} = \theta(1 - \phi)$

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The particular non-linear mapping function is based on two parameters, the users waist orientation and his or her distance from the back (i.e. front wall) of the Cave. The distance parameter is used to determine how much screen real estate is available to the user. The closer to the front wall of the Cave, the more screen real estate the user has to work with. The mapping function is a scaled 2D Gaussian as shown in the slide where σ_1 is a height constant, σ_2 is a steepness constant, and L is a normalization constant. The new viewing angle is simply the amplified viewing angle subtracted from the old viewing angle.

Amplified Non-Linear Rotation for VE Navigation (3)



$$\sigma_1 = 0.57$$

$$\sigma_2 = 0.85$$

$$L = 30$$

$$\mu = \pi$$

SIGGRAPH
2001 EXPLORE INTERACTION
AND DIGITAL IMAGES

The figure in the slide shows a graphical depiction of what the scaled 2D Gaussian mapping function is doing given the constants shown on the right. As d increases the Gaussian bump shifts to the left indicating a higher degree of amplification is applied to the current viewing angle. Therefore, as the user moves farther away from the back of the Cave, virtual world rotation will increase since it will take more rotation amplification to see a full 360 degrees.

Non-Linear Translation for VE Navigation (1)

- **Users lean about the waist to move small to medium distances**
 - users can lean and look in different directions
- **Users can also lean to translate a floor-based interactive world in miniature (WIM)**
 - Step WIM must be active
 - user's gaze must be 25 degrees below horizontal

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The second non-isomorphic technique we will discuss is a non-linear translation technique for navigating in virtual environments. The technique uses a leaning metaphor in which the user leans about the waist in order to navigate small to medium distances throughout the environment. In addition a user can translate a floor-based world in miniature (WIM) which allows for large scale navigation (more details on the floor-based world in miniature will be discussed in the Bringing 2D Interfaces to 3D Worlds lecture).

Non-Linear Translation for VE Navigation (2)

- **Leaning vector L_R is the projection of the vector between the waist and the head onto the floor**
 - gives direction and raw magnitude components
- **Navigation speed is dependent on the user's physical location**
 - Leaning sensitivity increases close to a boundary
- **Linear function - $L_T = a \cdot D_{\min} + b$**
- **Mapped velocity - $v = \|L_R\| - L_T$**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

In order to calculate the user's translation at each cycle of the simulation loop, we first obtain the raw direction and magnitude components from the leaning vector which is the projection of the vector between the user's waist and head onto the floor. Second, we find the minimum amount of lean the user has to perform to obtain a translation, which is dependent on the minimum distance between the user's position and a physical boundary in the leaning direction. We choose this approach based on the observation that users need to lean less when they are closer to a physical boundary. Thus, leaning is most sensitive when users cannot physically move any farther in a given direction. The mapped velocity is then found using the magnitude of the leaning vector and the minimum leaning value.

Non-Linear Translation for VE Navigation (3)

- Navigation speed is also dependent on the user's head orientation with respect to the vertical axis
 - especially useful when translating the floor-based WIM
- Mapping is done with a scaled exponential function

$$F = \alpha \cdot e^{-\beta |H \cdot V_{up}|}$$

- Final leaning velocity is

$$v_{final} = F \cdot v$$

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Once we obtain the mapped velocity, a second non-linear mapping function is applied to the user's translation velocity based on the observation that users tend to focus their gaze on the place they wish to go even when this location is moving towards them. In cases where objects are generally lower than the user's head height, a correlation of the movement rate to the user's head orientation with respect to the vertical axis allows for smooth deceleration when the user reaches his/her destination. Therefore, an exponential mapping function (alpha is the maximum speed factor and beta defines the steepness of the exponential curve) is applied to the mapped velocity to obtain the final translation velocity. This double mapping scheme works especially well for navigating the floor-based world in miniature.

Note that more details on this technique and the amplified rotation for navigation can be found in LaViola's paper, "Hand-Free Multi-Scale Navigation in Virtual Environments", included in the papers section of the course notes.

Conclusions

- **Non-isomorphic interaction techniques can be quite powerful**
 - faster for rotation tasks
 - allow for smoother transitions
 - fine tune techniques
 - navigate with hardware limitations
- **Still an active area for new discoveries**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Non-isomorphic interaction techniques can be quite powerful and are an important tool for interfaces designers to have at their disposal. They are still an active area of research and more work needs to be done to develop new mapping functions and evaluate how they affect user performance.

General References:

Poupyrev, I., S. Weghorst, and S. Fels. "Non-Isomorphic 3D Rotational Techniques", ACM CHI'2000, 540-547, 2000.

LaViola, J., D. Acevedo, D. Keefe, and R. Zeleznik. "Hands-Free Multi-Scale Navigation in Virtual Environments", In Proceedings of the 2001 Symposium on Interactive 3D Graphics, 9-15, 2001.



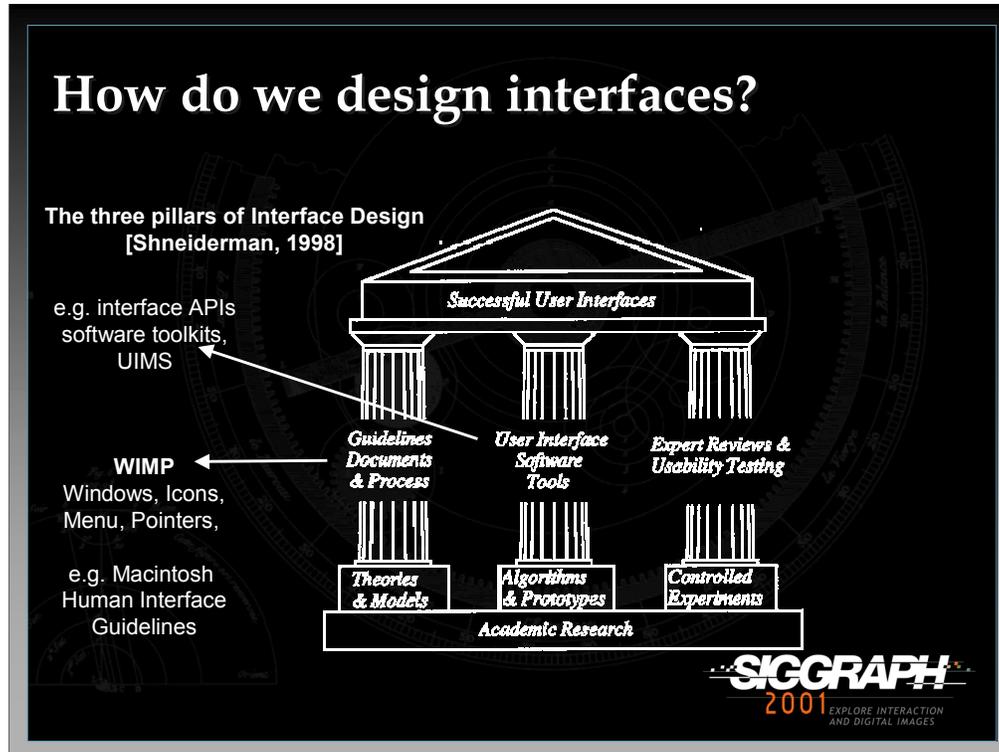
Ivan Poupyrev, Ph.D.
Interaction Lab, Sony CSL

E-mail: poup@csl.sony.co.jp

WWW: <http://www.csl.sony.co.jp/~poup/>

Address:
Interaction Lab, Sony CSL
Takanawa Muse Bldg.,
3 – 14 – 13 Higashigotanda
Shinagawa-ku, Tokyo 141-0022
Japan

How do we design interfaces?



Because 3D user interfaces are special types of human-machine interfaces, the results of many years of research and development of traditional 2D interfaces can be also applied to the design of 3D interfaces.

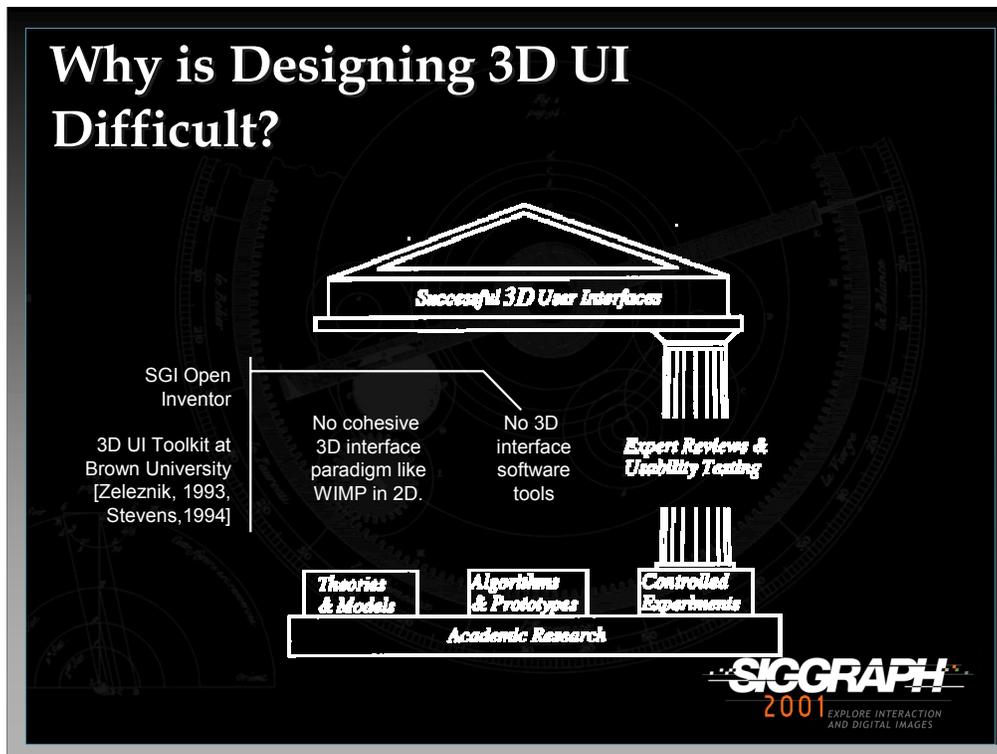
These are the three pillars of successful user interface design, according to Shneiderman (1998): *guideline documents*, *user interface software tools* and *expert review and usability testing*. An example of the guidelines that are often used in designing 2D interfaces is the Macintosh Human Interface Guidelines, which outline the basic elements of the user Macintosh 2D interface, their functionality, purpose, layout, and visual appearance (Apple, 1992). These and other similar 2D interface design guidelines provide designers with basic building blocks of the user interface. Thus, interface designers do not have to invent user interfaces themselves, but can construct interfaces out of instances of icons, menus, dialog boxes, windows and others interface elements, as well as assign them various properties, names and functionality.

Furthermore, interface designers and developers do not even have to implement these basic interface elements: the user interface API (i.e. Application Programming Interface) provides access to libraries of already implemented behavior and functionality of the interface elements, which has become a standard part of the operating system. High-level interface design tools such as graphical editors allow designers to “draw” an interface for application, while even higher level tools such as UIMS (user interface management systems), provide ever more powerful tools in designing interfaces.

-continued on the next page

Certainly, even with the help of all these tools, designing high-quality interfaces still remains a complex and challenging task, requiring multiple iterations and usability studies to evaluate and refine designed interfaces (a third pillar of the interface design in Shneiderman's diagram).

While there has been a lot of criticism of the dominant desktop WIMP paradigm (e.g. Norman, 1999), it cannot be denied that, in spite of all their shortcomings, desktop graphical user interfaces have been a major step toward interfaces that can be effectively used by large numbers of users across different computing platforms.



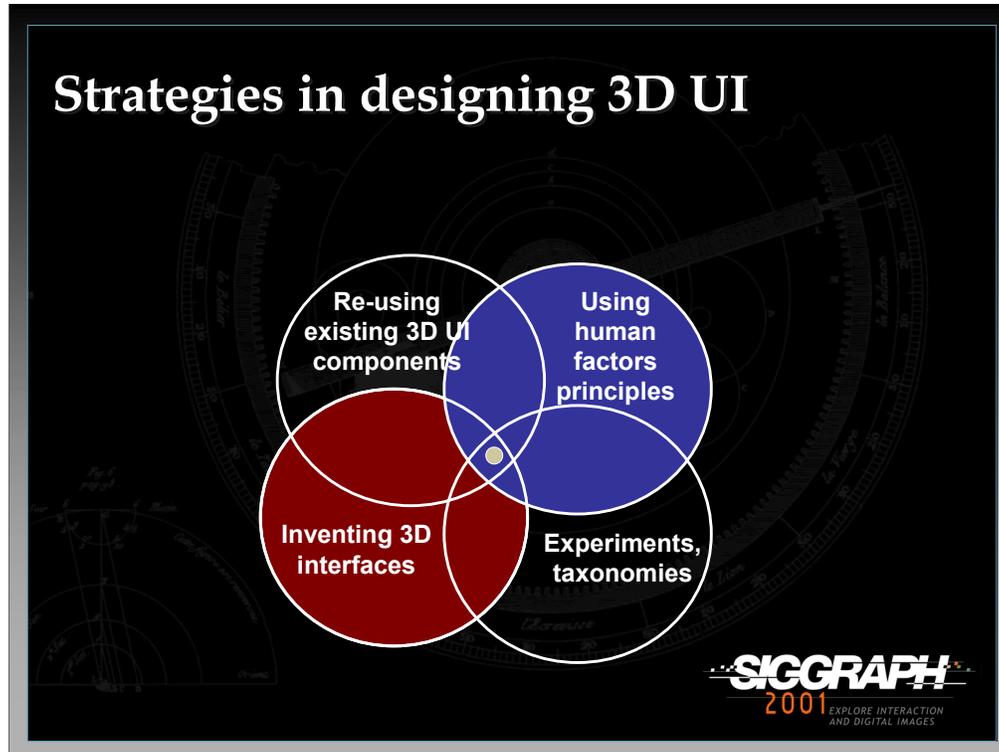
Designing 3D interfaces is still an art because first, there no cohesive 3D interface paradigm exists. What are the most basic classes of elements for 3D user interfaces? How do they relate to each other? There have been many 3D interaction techniques reported in the literature, some of them with guidelines for their use. However, it is not clear how they all relate and compare to each other, or how we should approach design of complex interaction sequences to do complex tasks. Consequently, there are few if any 3D interface design guideline documents that the designer of 3D user interfaces can rely on.

Second, there are currently no tools to support the design of 3D user interfaces, beyond the most simple, for example in Open Inventor. Currently, if designers need to use certain interaction techniques or tools they must either implement them themselves from scratch or invent new techniques. As a result designing interactive 3D user interfaces is a very time consuming task. In addition the produced interfaces are rarely formally evaluated and are usually designed mostly on the basis of designer intuition and common sense. Consequently, today's 3D interfaces are incompatible from application to application: each has its own look and feel.

One of the reasons for this is that the design space in 3D user interfaces is significantly larger then in 2D and large portions of it remains unexplored. 3D interface technology is still rapidly developing, and the new input and display devices, interaction techniques that appear require consequent reevaluation. Furthermore, it is not uncommon that for certain application tasks or devices there have been no interaction techniques constructed, i.e. the design space of 3D user interfaces still has many empty spots.

-continued on the next page

In conclusion, the design of 3D interfaces is currently based on a mixture of intuition, common sense, informal rules of thumb, previously reported or ad-hoc designing techniques, and few general human factors guidelines.



Today there are basically four major “pillars” in designing 3D user interfaces. First of all, we can reuse some of the interaction techniques developed before. The previous lectures in this tutorial have presented some of the interaction techniques reported in the literature.

Second, the human factors literature, related both to general principles of human-machine interaction as well as to specific principles of interaction with computers, is also highly relevant in the design of 3D user interfaces.

Third, the design of complex interaction sequences often require developers either to invent new interaction techniques or to adopt existing interaction techniques. This is because 3D interaction is a rapidly developing field and new input and output devices as well as new applications that employ 3D input are constantly being developed.

Finally, the design can be guided by some of the 3D user interface taxonomies and results of experimental studies reported in the literature. Some of the taxonomies and systematic design techniques will be covered in the other lectures of this course.

Lecture Goal and Outline

- **Discuss general strategies in designing 3D UI**
 - Designing for humans
 - Techniques for inventing 3D user interfaces
 - Realism in 3D user interfaces
 - Magic in 3D user interfaces
 - Advantages and disadvantages
 - Video examples of 3D interface design

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

In this lecture I will discuss some of the informal and general approaches for designing 3D interfaces.

The lecture starts with discussing some of the basic human factor principles that can help to design better 3D user interfaces. The area of human factors engineering is vast and any in-depth discussion is far beyond the scope of this lecture. However, a few of the often used and reported principles will be briefly discussed.

The lecture then continues by presenting simple techniques for inventing 3D user interfaces. In order to present these principles in a more or less cohesive manner, I will separate them into two categories: first are methods based on the “realism” (or isomorphism) in 3D UI design, an approach which tries to borrow ideas from the real world. The second category is “magic” or non-isomorphism. In this approach we are trying to design interfaces that are significantly different from the real world and that allow the user to interact with 3D computer graphics environments in a very different manner than in a real world.

Methods and ideas that will be discussed in this lecture are general in the sense that they apply not to a single task or application, which was the case of the techniques that we talked about before, but to any interaction technique, interface or 3D system. Most of the design principles discussed here are informal, based on rules of thumb, esthetics, and stealing from other areas of human activity. Nevertheless, many of today’s successful 3D interfaces have been designed based on these ideas.

Designing for humans: basic principles

- **Basic principles (Shneiderman, 1998)**
 - Simplicity, reduction of short-term memory load
 - Consistency of interface syntax and semantics
 - Feedback of operation completion
 - Error prevention, handling
 - Aesthetic appeal



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Most of the interface design principles from human factors research can be directly applied to designing 3D user interfaces. Simplicity, consistency, feedback from operations, error prevention and aesthetic appeal are as important in 3D interaction as in any other human-machine interfaces.

Designing for humans: adding constraints

- **Geometrical Constraints**
 - Collision detection
 - Reducing degrees of freedom
 - Snapping, gravity functions (e.g. Bier, 1990)
 - Precise alignment, scene creation
 - Intelligent constraints (Smart Scenes, Polyshop)

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The simplest example of constraints is using collision detection: the users freedom is limited by not allowing them to go freely through virtual objects, which in many cases makes interaction much easier, especially in navigation techniques.

Constraints have been used in 3D user interfaces in two main forms. First, they were used to reduce the number of degrees of freedom to make manipulation simpler. For example, an object can be constrained to only move on the surface of a plane, making it positioning simpler.

Second, constraints we used to snap objects to a 3D grid or special guiding objects, e.g. surfaces, lines and planes with which manipulated objects aligns (e.g. Bier, 1990). Snapping can make selection and object arrangement significantly easier.

Designing for humans: two handed operation

- **Two-handed manipulation**
 - Transfer of everyday skills
 - Performing two tasks in parallel
 - Symmetric manipulation
 - Two hands do a single task
 - Asymmetric manipulation



From Hinckley et al., 1997

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Using both hands while interacting with computers allows us to transfer our everyday manipulation skills into interaction with virtual environments which makes it easier and more effective. That is why, two-handed or bi-manual input has been investigated extensively in 2D interfaces (e.g. Buxton et al. in 1986). In 3D interaction two-handed input has also been successfully used to design compelling 3D user interfaces (e.g. Mapes and Moshell, 1995).

There are two basic ways to incorporate two hands in 3D interaction. The first is *symmetric* bi-manual manipulation, where each hand can be used to perform different, separate tasks. Interaction is symmetrical in the sense that input from both hands is equal, for example typing on a keyboard.

A second approach is to allow the user to use both hands to perform a single task, for example selecting from a hand held menu or rotating an object with one hand while fixing the center of rotation with the other. In this case of bi-manual input the use of the hand is asymmetrical in the sense that each hand assumes a certain role that depends on the action of the other hand. Hinckley (1997) demonstrated that in the cooperative two-handed manipulation the left (non-dominant) hand defines a general spatial frame of reference for precise actions of the right (dominant) hand. This property can be explicitly used to design interesting interaction techniques for object manipulation, for example the Voodoo Dolls technique (Pierce et al., 1999).

Designing for Humans: Feedback

- **Feedback**
 - Self-regulation of actions
 - Multidimensional feedback
 - Tactile, haptic, visual and aural, etc.
 - Spatial and temporal correspondence
 - Stimulus-response compatibility



Haptic interface,
Northwestern University

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Feedback plays crucial importance in designing user interfaces. Our ability to self-regulate body movements, e.g., manipulating objects, depends on spatial and temporal correspondence between a large variety of sensory feedbacks: visual, tactile, kinesthetic, proprioceptive and others. If the 3D user interface response, e.g., visual feedback, conflicts with kinesthetic or proprioceptive feedback produced by the human motor system, then the user performance degrades (e.g. Smith and Smith, 1987). Hence, the quality of 3D user interfaces depends on whether they preserve *compliances* between the multiple-dimensions of sensory feedback, i.e., a stimulus-response (S-R) compatibility (Fitts, 1953).

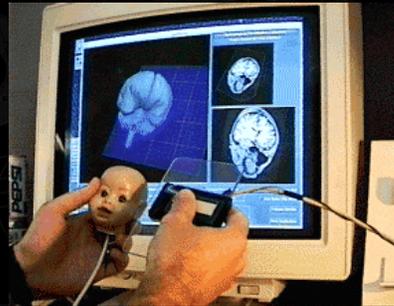
Designing for Humans: Feedback with passive props

- **Tracked physical objects resembling virtual objects**

- “props” Hinckley
- “near field haptics” Brooks
- “tactile augmentation” Hoffman

- **Active props: props with functionality**

- Virtual Tricorder
- Virtual Notepad



Props-based interface for 3D
Neurosurgical Visualization
Hinckley, 1994

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

One popular technique in 3D user interface design is to consider and control the physical shape of input devices, providing the user with passive tactile feedback. The main idea here is to match the shape and/or appearance of a physical object with a virtual one. The term that is used to refer to this technique is “*passive physical props*” and they were first introduced by Hinckley when he described a 3D interface for visualizing and interacting with neursurgical data (Hinckley et al., 1994). The input device in his interface was a doll’s head fitted with a 6DOF sensor. By manipulating the dolls head, the user was able to quickly and reliably relate the orientation of the input device to the orientation of the brain data on the screen, resulting in efficient and enjoyable interaction.

The interface designed by Hinckley was non-immersive, in immersive environments the passive props can also be spatially registered with virtual objects providing inexpensive physical feedback to the user. Hoffman refers to the technique as “tactile augmentation” (Hoffman et al., 1998).



-continued on the next page

Virtual tricorder by Wloka et al. 1996

Extending this technique, we can design *active physical props* by matching shape and functionality of input devices with a corresponding virtual object. Active physical props were first introduced by Wloka (1995), where a physical mouse was registered with a virtual mouse of the exact same shape, allowing the user to easily interact with it while immersed in a virtual environment. Another example of an active physical prop is the Virtual Notepad, where a virtual tablet is matched with a real pressure-sensitive tablet, allowing the user to write while immersed in VE (Poupyrev et al, 1998).

Designing for Humans: Feedback with passive props

- **Advantages**

- Haptic and tactile feedback
- Increases presence (Hoffman, 1998)
- Common reference frame (Hinckley, 1994)
- Ease of learning
- Direct control of realism
 - e.g. treatment of phobias

- **Disadvantages**

- Tracking multiple objects
- Does not increase performance (Hinckley 97, Ware and Rose 99)



Using props in treatment of spider phobia Carlin, Hoffman et al., 1994

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Using passive and active physical props is an extremely useful design technique for 3D user interfaces. Props allow us to add inexpensive physical and tactile feedback, significantly increasing presence for immersive environments (Hoffman et al., 1998) and establishing a common frame of reference between the device and desktop 3D user interfaces. The introduction of tactile augmentation allows us to explicitly control the realism of virtual environments, which can be useful in such applications as the treatment of phobias (Carlin et al. 1997).

The disadvantages of props are that they require tracking of multiple physical objects, which might be expensive. Also, experimental studies done so far have not shown any improvement in user performance for motor tasks when using props (Hinckley 97, Ware and Rose 99).

Inventing 3D UI

- **Realism (or isomorphism)**
 - Borrowing from real world
- **Magic (or non-isomorphism)**
 - Deviating from the real world and introducing artificial, magic techniques
- **Continuum between realism and magic**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

While human factors principles offer important design guidelines for 3D interface design, they do not really suggest how we can create compelling 3D interfaces or invent effective techniques. In this section, I survey some of the informal, rule-of-thumb approaches that have been taken in designing 3D user interfaces.

The approaches that I will discuss are categorized into two categories. The first includes approaches based on “realism” (or isomorphism), an approach that tries to borrow ideas for 3D interface design from the real world. The second is the “magic” approach, or non-isomorphism, in which we are trying to design interfaces that are significantly different from the real world and allow the user to interact with 3D computer graphics environments in a very different manner than in a physical world.

While this categorization is useful, it is not very strict – usually there is a continuum between realism and magic.

Realism: copying the real-world

- **Examples**

- Virtual hand manipulation
- Physical walking on treadmills

- **Advantages**

- + A must for simulation applications
- + User is familiar from everyday experience
- + Implemented on the basis of designer intuition

- **Disadvantages**

- Limitations of technology do not allow exact realism
- Introduces limitations of the physical world into the virtual world



Virtual hand from Poupyrev (1996)

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The basic approach to design 3D interfaces is to simply imitate the real, physical world as closely as possible. This approach is important for all simulation applications, such as training, battle field simulation, some entertainment applications, and evaluations of the usability of complex human-controlled mechanisms such as cars and tractors.

The advantages of this approach is that the user already knows how to use the interface from everyday experience, and an interface can be implemented either on the basis of the designer's intuition or clearly specified technical design requirements, such as in simulation applications.

The problem, however, is that the simulation is never exact due to the limitations of the technology. In non-simulation applications this approach introduces the same limitation as we have in the real world, which might be annoying and inefficient for the user. Furthermore, even in a simulation application there are often tasks that do not directly relate to the simulation itself, e.g. system control, and require use of 3D interaction techniques.

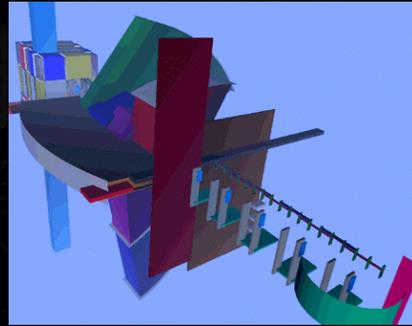
Realism: Adapting from other disciplines (I)

- **Leaving spaces, tools and media**

- architecture and movies
 - Campbell 96, Tanney 98
- widget and tool metaphors
- virtual vehicle metaphors

- **Natural gestures**

- pulling oneself along the rope to move forward (Smart Scene, Multigen Inc.)
- breathing as in scuba diving to go up or down (Osmose, by C. Davis)



Dace Campbell, 1996,
Virtual Architecture

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Instead of mimicking a real world we can steal and adopt ideas and/or existing artifacts from the real world. Indeed, movies and architecture has been a source of inspiration for much of VE design (e.g. Campbell 1996, Herndon et al., 1994). The virtual vehicle metaphor has been probably one of the most used techniques for navigation. Virtual widgets and tools have been often adopted from real-world physical tools and objects: for example in the dVise system from Division Inc, a lamp widget was used to set up lighting and an egg widget was used to create new objects.

Another way to adapt a real world for the virtual is to borrow natural physical gestures to perform interaction tasks. For example, in the Smart Scenes by Multigen Inc. (see also Mapes, 1995) the user moves in the environment by pulling himself or herself along an invisible rope. An even more radical method was used in the Osmose environment, where the user navigated by using breathing and balance control, a technique borrowed from the scuba diving technique of buoyancy control. The user is able to float upward by breathing in, to fall by breathing out, and to change direction and by altering the body's center of balance. The intent was to create an illusion of floating rather than flying or driving (<http://www.softimage.com/Projects/Osmose/>).

While borrowing from the real world, these techniques do not simply mimic but rather adopt real world tools and gestures, which can make the interface rather intuitive.

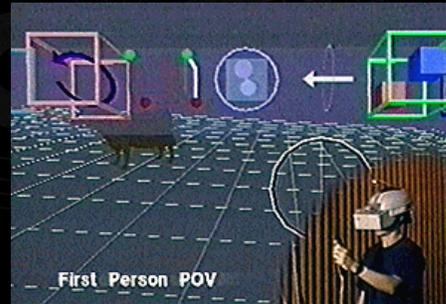
Realism: Adapting from other disciplines (II)

- **Advantages**

- + it's already done
- + search for solutions around one
- + experience transfer
- + can be very easy to understand

- **Disadvantages**

- analogy is never exact
- difficult to find analogy for abstract operations
- when is it really effective?



Widgets by Mine et al. , 1996

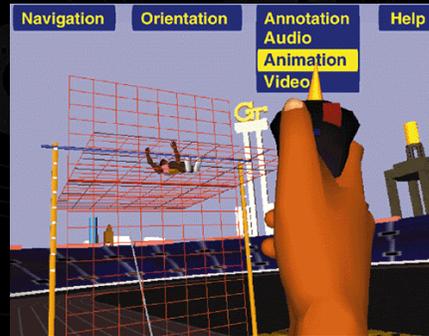
SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The advantages of this approach are that there is a large number of objects that we can adopt for VR interaction in almost any task, the user can transfer his or her own real world experience to virtual worlds, and analogies are usually easy to understand.

The disadvantages are that any analogy is never complete and it is usually difficult to find good analogies for abstract operations. For example in the dVise system from Division an egg widget was used to create new objects, a metaphor that is not so transparent. Finally, it is not clear if the adaptation is effective or not unless we conduct extensive experimental studies.

Realism: Adapting from other 2D interfaces (II)

- **Examples:**
 - 2D menus
 - Drawing and pointing
- **Advantages:**
 - + 2D interfaces have been thoroughly studied
 - + many people are familiar with 2D user interfaces
 - + some tasks are easier to accomplish in 2D
- **Disadvantages:**
 - Not always appropriate



Virtual Athletic Venue, GVU

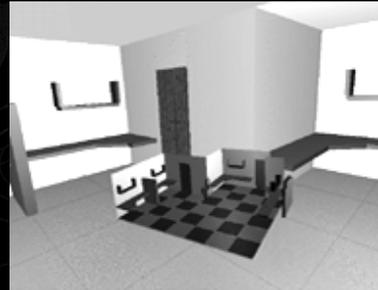
SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

When we cannot borrow from the real world, why don't we borrow from 2D user interfaces? There have been quite a few attempts to do this, usually for system control tasks, menus and symbol control (e.g. Bolter, 1995). The major advantages are that 2D user interfaces have been thoroughly studied and today's users are quite familiar with 2D interfaces. The problem is that the 2D interfaces are not always appropriate for 3D interaction tasks simply because they have not been designed for 3D interfaces.

Magic: Stretching, extending real-world metaphors

• Examples

- *Go-Go arm stretching technique*: body centered arm length amplification (Poupyrev, et al. 1996)
- *World-in-Miniature*: hand centered miniature 3D world map (Stoackley, et al. 1995)

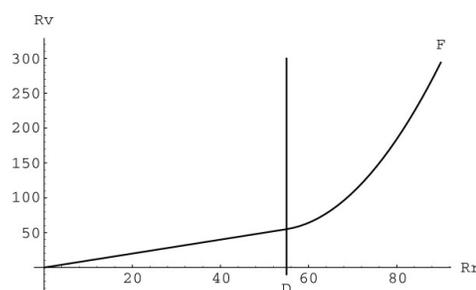


World-in-miniature

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

One of the approach of magic in 3D use interfaces would be to extend the user ability or change the geometrical properties of the real world. Two examples considered here are the Go-Go techniques (Poupyrev, et al. 1996) and World-in-Miniture (Stoackley, et al. 1995). The **Go-Go** technique, flexibly extends the virtual hand technique reaching distance by using a non-linear mapping function applied to the user's real hand extension. The space around the user is split into two concentric regions. While the user's the real hand is within the first closest region around the user, that is, the distance to the hand is smaller then some threshold distance D , the mapping is one-to-one and the movements of the virtual hand correspond to the real hand movements (see figure below). However, as the user extends her hand further than D , the mapping becomes non-linear and the virtual arm "grows" allowing the user to reach and manipulate distant objects.

-continued on the next page



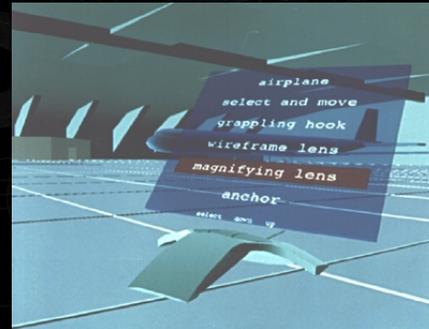
Mapping function for the Go-Go
(Poupyrev, et al. 1996)

The **World-In-Miniature** (WIM) technique (Stoakley et al. 1995) provides the user with a miniature hand-held model of the VE, which is scaled down using some constant coefficient (see figure below). The user can then indirectly manipulate virtual objects by interacting with their representations in the WIM.

The WIM technique is a powerful technique allowing easy object manipulation both within and outside of the area of user reach. It also can combine navigation with manipulation since the user can easily move his or her own representation on the WIM. The downside of the technique is that scaling large environment results in very small representations of objects in the WIM, so accurately manipulating small objects might be difficult. A technique that can choose the part of the environment within the WIM might overcome this problem.

Magic: Cultural Clichés and Metaphors

- Flying carpet
- Voodoo Dolls (Pierce, 1999)
 - Indirect interaction through hand-held proxies (dolls)
- Virtual Tricorder (Wloka, 1995)
 - Universal control device for virtual reality

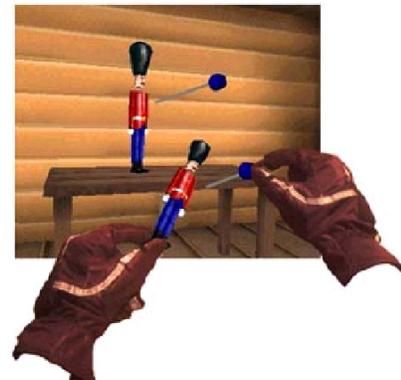


Virtual Tricorder

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Cultural Clichés and metaphors, such as flying carpet, can also suggest an interesting approaches in designing 3D user interfaces. For example a **Voodoo Dolls** technique (Pierce, et al. 1999) is a two-handed interaction technique for manipulating objects at a distance in immersive virtual environments. The technique combines and builds upon a number of other techniques, such as Image Plane (Pierce et al., 1997) and WIM (Stoakley et al., 1995). Voodoo Dolls uses a couple of pinch gloves to allow the user to switch seamlessly between different modes of manipulation. It aims to provide an easy method of interacting with objects of widely varying sizes and at different distances. The technique is based on several ideas. First, to start object manipulation the user dynamically creates dolls: temporary, miniature, hand-held copies of objects. Similar to the WIM technique, the user can interact with objects in the environment by manipulating these dolls instead of directly manipulating the objects so that manipulated virtual objects can be at any distance, size and state of occlusion.

Second, the technique allows the user to explicitly and interactively specify a frame of reference for manipulation. The doll that the user holds in the non-dominant hand represents a stationary frame of reference, and the corresponding virtual object does not move when the the user moves this doll.



Magic: Advantages and Disadvantages

• Advantages:

- + easy to understand if you know the metaphor
- + usually they are very enjoyable
- + many metaphors are available
- + need not to be learned

• Disadvantages:

- the metaphors can be misleading
- the metaphors are often rooted in culture
- It is difficult to come up with good magic metaphor

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Magical approach does not try to incorporate properties of the physical world into the virtual environment, but rather extends them by inventing “magical” interfaces . All of these techniques are based on certain “magical” metaphors and are very easy to understand if one knows the metaphor. Many metaphors are available that can be used and they do not needed to be learned if the user already knows about them. Thus the resulting interface can be very easy to learn and used right away.

A problem with this approach is that metaphors are never complete, and they are often misleading, especially magical ones. Metaphors are rooted in culture: indeed if one has never heard about a flying carpet then the metaphor might not work. Finally, it is not that easy to find effective and compelling magical metaphors. However, if one is found it can provide a very enjoyable user interface.

Conclusions

- **Designing 3D interfaces today is more art than guided design**
 - Re-use of previous interaction techniques
 - Basic human factors principles
 - Inventing interaction techniques
 - Usability evaluations of designs
- **The guidelines and methods are slowly emerging**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Bibliography

For most of the references see reference section of the notes. Below are only those references that are not in the main bibliography.

- Apple Computer, I., *Macintosh Human Interface Guidelines*. 1992: Addison-Wesley Publishing Company. pp. 384.
- Carlin, A., Hoffman, H., Weghorst, S., Virtual reality and tactile augmentation in the treatment of spider phobia: a case report. *Behavior Research and Therapy*, 1997. 35(2): pp. 153-159.
- Buxton, W., Myers, B., A Study in Two-Handed Input. *Proceedings of CHI*. 1986. ACM. pp. 321-326.
- Fitts, P., Jones, R., Compatibility: spatial characteristics of stimulus and response codes. *Journal of Experimental Psychology*, 1953 (46).
- Shneiderman, B., *Designing the user interface: strategies for effective human-computer interaction*. 3 ed. 1998: Addison-Wesley. pp. 638.
- Smith, T., Smith, K., Feedback-control mechanisms of human behavior. In *Handbook of human factors*, G. Salvendy, Editor. 1987, John Wiley and Sons. pp. 251-293.



Bringing 2D Interfaces into 3D Worlds

Joseph J LaViola Jr

Ph.D Candidate

Brown University,

Department of Computer Science

Providence, Rhode Island

Lead Consultant and Founder

JJL Interface Consultants, Inc.

http: <http://www.cs.brown.edu/people/jjl>

email: jjl@cs.brown.edu

Goals and Motivation

- Bring strengths of 2D input into 3D applications
- Develop seamless integrations between 2D and 3D
- Examine classical approaches
- Extend 2D/3D beyond classical techniques

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

In this lecture, we will discuss the advantages of 2D interaction and how we can use various 2D techniques in 3D applications. We will examine both classical approaches and state of the art research results.

Lecture Outline

- **Strengths of 2D and 3D interfaces**
- **Seamless integration**
- **2D/3D Interface Taxonomy**
 - Virtual Notepad
 - ErgoDesk
 - Virtual Palette
- **Go beyond traditional approaches**
 - StepWIM
 - TULIP

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

In the first part of the lecture, we will examine the strengths and weaknesses of 2D and 3D interfaces and how they can be seamlessly integrated as one interface to 3D graphics applications and virtual environments. Second, we will briefly develop a 2D/3D interface taxonomy based on previous work in the area and examine some interfaces that fall into this categorization. Finally, we will look at a couple of interfaces that go beyond our taxonomy in the sense that they bring 2D interaction concepts into 3D applications in unconventional ways.

2D Interaction

- **Advantages**

- provides a sense of feedback
- very accurate
- some operations that are 3D in nature are more easily done with a 2D input device (e.g. object selection)
- picking objects is much easier in two dimensions

- **Limitations**

- manipulating 3D objects
- have to add 3rd dimension in unconventional and unnatural ways
- WIMP

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

2D interaction techniques have both advantages and disadvantages as shown in the slide.

3D Interaction

- **Advantages**

- more natural for object manipulation once the object is taken
- take advantage of 3D hand gestures and postures
- stereoscopic vision

- **Limitations**

- very difficult to write and annotate
- difficult to pick and place objects accurately

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

3D interaction techniques have both advantages and disadvantages as shown in the slide.

Bringing 2D and 3D Together

- **Goal: Let's take the advantages from each type of interaction and bring them together to form a more usable interface**
- **Broaden the application space**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

By taking advantage of the benefits of both 2D and 3D interaction techniques and metaphors, we can create interfaces for 3D applications that are easier to use and more intuitive for the user. The key research issue is how to combine these two input styles in a seamless manner and to determine whether a particular task is better suited for either 2D or 3D interaction so we can maximize user performance.

Seamless Integration

- **Critical component**
- **Requires both physical and logical integration**
- **Do not want the user to work hard to change modes**
- **Tools should know what interaction technique they are used for**
 - a device should know whether it is used for 3D interaction or 2D interaction based on context

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The seamless integration of the 2D and 3D interface techniques in a 3D application is a critical design consideration from both a logical and a physical perspective. Physical integration is important because we do not want to make it difficult for the user to switch between 2D and 3D devices. Logical integration is also important because we want the devices used in the application to know whether they are used for either 2D or 3D interaction. This knowledge helps to reduce the user's cognitive load.

2D/3D Interface Taxonomy

- Based on display surface interaction
- Taxonomy
 - direct
 - hand-held indirect
 - hand-held direct

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

There is a pattern in the 2D interface in 3D application literature which leads to a simple taxonomy based on how the user performs the 2D operations. In general, there has to be a some type of surface with which the user can interact on. The first category in the taxonomy is *direct display surface interaction*. Any display surface with allows the user to perform 2D operations falls into this category. Desk-based displays which allow the user to draw on the display surface are a good example. The second category is *hand-held indirect display surface interaction*. This category includes applications which require the user to hold a pad, whether transparent or opaque, in order to perform 2D operations. A classic example is in virtual environments where users must wear HMDs and cannot see the physical world. A virtual display is presented to the user in the virtual world and is correlated with the physical pad. Finally, the third category is *hand-held direct display surface interaction*. This category includes applications which use hand-held displays or computers that allow the user to interact in 2D on their display surfaces.

In the next few slides we will examine a few examples of 2D/3D interfaces that fall into these categories.

Direct Display - ErgoDesk

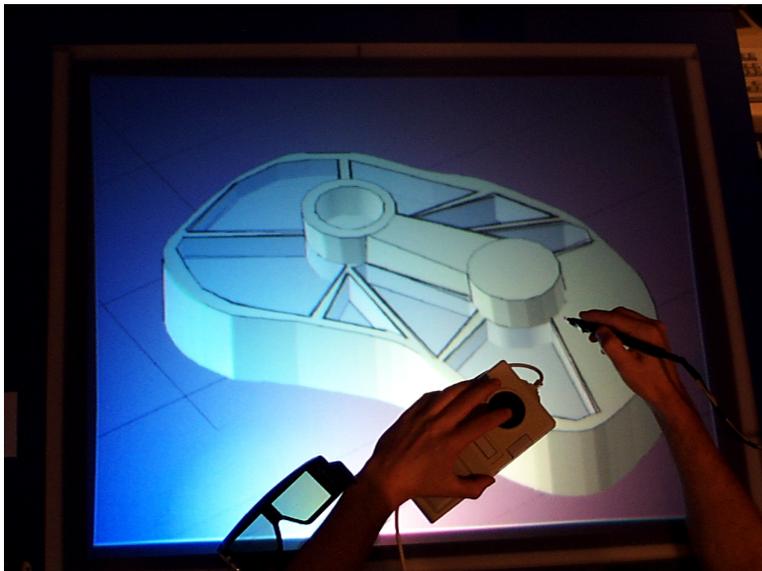
- 3D modeling application
- 2D interaction on display surface
- Based on Sketch
- Allows users to create, edit, view and manipulate 3D models



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

ErgoDesk is an example of a 2D/3D interface where the user interacts in 2D directly on the display surface. The 2D component of the ErgoDesk application is based on the Sketch conceptual modeling system which uses only a three button stylus (no menus or 2D interface widgets are used). Sketch interprets lines drawn by the user on the image plane of a 3D view as operations and parameters. These operations include primitive creation, primitive manipulation, and camera manipulation. Gestures that create primitives provide enough information to select which primitive to create, its dimensions and its place in 3D. Creating a cube, for example, requires the user to draw 3 gesture lines one for each of the principle axes, each line meeting at a single point. The cube is generated with its length, width, and height corresponding to the three gesture lines and its place in 3D based on the intersection point. Primitives such as cylinders, cones, pyramids, and extrusions can also be instantiated. The primitive manipulation interface allows for automatic object constraint by gesturally drawing a motion constraint over the object before manipulating it. For example, to constrain an object's movement to a given axis, a straight line is drawn indicating what axis to constrain the object to, and when the user moves the object it will only move along that axis. Other gestures constrain objects to move along surfaces, rotate around a given principle axis, or scale and deform to fit a new gesture contour.

-continued on the next page



References:

Forsberg, A., LaViola J., and Zeleznik, R. "ErgoDesk: A Framework For Two and Three Dimensional Interaction at the ActiveDesk." In the Proceedings of the Second International Immersive Projection Technology Workshop, Ames, Iowa, May 11-12, 1998.

Zeleznik, R.C., Herndon, K., Hughes, J. (1996) "Sketch: An Interface for Sketching 3D Scenes." Proceedings of SIGGRAPH'96, 163-170.

Hand-Held Indirect (1): Virtual Notepad

- Tool for writing in immersive environments
- Allows users to take notes and annotate documents



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The Virtual Notepad is an example of a 2D/3D interface where users cannot physically see the 2D device since they are wearing an HMD. The 2D device is tracked so a graphical representation of it is present in the virtual environment.

References:

Poupyrev, I., Tomokazu, N., Weghorst, S., "Virtual Notepad: Handwriting in Immersive VR". IEEE VRAIS'98, 126-132, 1998.

Hand-Held Indirect (2): Transparent Pad

- **Transparent prop for the Virtual Table**
 - tool and object palette
 - window tools
 - through-the-plane tool
 - volumetric manipulation



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The Transparent Pad is another example of a 2D/3D interface which utilizes a hand-held pad to perform 2D operations. In this case, the pad is transparent. The pad is tracked and graphics are projected on the primary display but appear as if they are on the surface and even above the pad.

References:

Schmalsteig, Dieter, L. Miguel Encarcacao, Zsolt Szalavari. "Using Transparent Props For Interaction with The Virtual Table." In Proceedings of the 1999 ACM Symposium on Interactive 3D Graphics, 147-154, 1999.

Coquillart, S. and G. Wesche. "The Virtual Palette and the Virtual Remote Control Panel: A Device and Interaction Paradigm for the Responsive Workbench." IEEE VR'99, 213-217, 1999.

Hand-Held Direct Displays

- **PDA's in Immersive VEs**
 - Watsen used PalmPilot in a CAVE-like device [IPT99]
 - provides camera, environment, and geometry controls
- **Wacom Tablet in the TAN-Cube**
 - too heavy
 - wires got in the way
 - has potential

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

There are only a few reported cases of using hand-held direct displays for performing 2D operations in 3D applications. One of the first used a PalmPilot to control camera, environmental, and geometrical parameters in a virtual environment. With better wireless technology and more light weight, portable display devices hand-held direct display interaction should gain more prominence in 2D/3D interface research.

References:

Watsen, Kent, Rudy Darken, and Michael Capps. "A Handheld Computer as an Interaction Device to a Virtual Environment." Proceedings of the Immersive Projection Technology Workshop, Stuttgart, Germany, May 1999.

Going Beyond the 2D/3D

Taxonomy

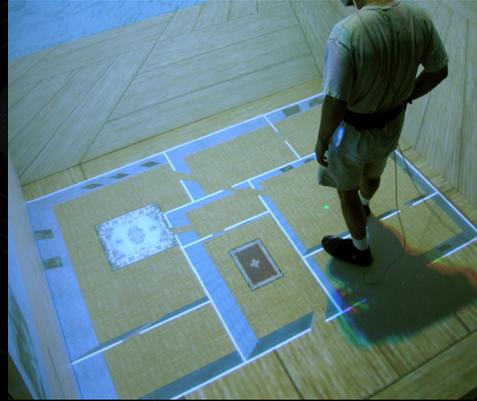
- Go beyond the 2D surface and hand approach
- Utilize traditional 2D concepts and extend to 3D interfaces
 - Step WIM – based on maps
 - TULIP – based on 2D menus

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Our 2D/3D interface taxonomy has a common theme in that the 2D interface component is derived from a surface and hand metaphor. In each category, the user interacts with a display surface of some kind with his/her hands to perform various 2D and even 3D operations. An interesting research question is whether we can go beyond our taxonomy and utilize traditional 2D interface concepts without the constraint of the 2D surface and hand approach. Of course the answer is yes; otherwise the lecture would be over. In the next few slides, we will look at two examples which go beyond our 2D/3D classification. The first is the Step WIM, which keeps the 2D surface concept but is a hands-free interface which utilizes body gestures and the user's feet. The second is TULIP, which allows the user to interact with the hands but removes the 2D surface constraint allowing for interaction in 3D space.

The Step WIM

- Miniature version of the world placed on the floor
- Motivated by Pausch and Stoakley's WIM
- Augmented roadmap
- Step WIM scales up around users feet
- Operations
 - invoking
 - navigating
 - dismissing
 - scaling



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The Step WIM is a interaction widget for quickly navigating through a virtual environment. It is a miniature version of the world placed underneath the user's feet and acts as an augmented roadmap. The user can either walk around the Step WIM to get a better understanding of the virtual world or navigate to a specific place by simply walking to a desired location in the WIM and invoking a scaling command, causing the Step WIM to animate, scaling up around the user's feet, thereby seamlessly transporting the user to the specified location.

Foot-based Interface

- **Toe and heel tapping**
 - “no place like home” metaphor
- **Developed interaction slippers**
- **Disambiguation of navigate and dismissal**
 - based on user gaze
 - derived from pilot studies



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

In order to invoke, navigate and dismiss the Step WIM, users can wear a pair of interaction slippers (slippers with an imbedded wireless mouse) which gives them the ability to perform toe and heel tapping. To invoke the Step WIM, users simply tap their toes together. Once the Step WIM is active, another toe tap will transport the user to a new location or dismiss the widget without navigation. Based on pilot studies, users tended to look down at the Step WIM when they wanted to navigate so disambiguation of the navigation and dismissal tasks are based on user gaze. This approach allows for two distinct operations to be mapped to one button press.

Body Gesture Interface

- **More fluid gesture/less invasive device**
- **Use waist tracker to detect upward bouncing gestures**
- **Algorithm**
 - first get user's initial waist height
 - monitor the waist tracker's position
 - check to see if the waist is above a height delta for a given amount of time

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Another way to use the Step WIM is based on body gestures. Using a waist tracker, a simple gesture recognition algorithm detects upward bouncing movements. When the user performs a bouncing gesture, the Step WIM is activated. Another bouncing gesture dismisses the Step WIM or transports the user to a new location, once again depending on user gaze.

Step WIM Scaling

- VEs may be too large to fit within user's walking area
- Scaling implicitly provides different levels of detail



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Since the virtual environment may be too large to fit within the user's walking area, the Step WIM can be scaled to varying sizes. This scaling implicitly provides different levels of detail.

Foot-based Scaling

- **Heel click toggles Step WIM scaling mode**
- **Center of scale is user's initial "location" in WIM**
 - maintain position within the WIM
- **Walking forward – closer look at the world**
 - Step WIM grows larger
- **Walking backward – gain perspective**
 - Step WIM grows smaller

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

When using the interaction slippers, a heel click toggles in and out of Step WIM scaling mode. When the user walks forward from his or her initial position (the position defined with the heel click), as if to take a closer look at the world, the Step WIM grows larger. When the user walks backward from his or her initial position, as if to gain perspective, the Step WIM grows smaller.

Body Gesture Scaling

- Avoid cue conflict of “walking in place”
- Holding a crouching gesture increases Step WIM size
- Holding a bouncing gesture decreases Step WIM size
- Center of scaling is projection of user’s *waist*
- Gestures must be held longer than the bounce time threshold
 - distinguishes between scaling and activation/dismissal

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

With the body gesture-based interface, holding a crouching gesture increases the Step WIM size, while holding a bouncing gesture decreases the size of the Step WIM.

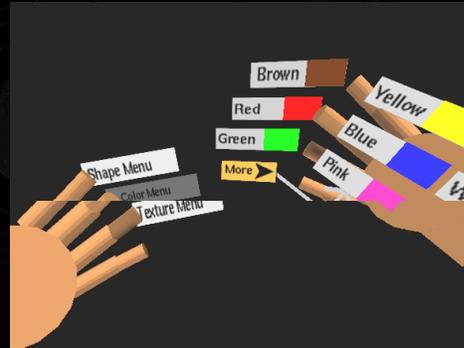
In general, the Step WIM represents a 2D concept (the concept of a map) that has been incorporated into a 3D interface for navigation. This interface technique goes beyond our 2D/3D taxonomy by removing the hand component of the 2D surface and hand metaphor. More information on the Step WIM can be found in the paper entitled, “Hands-Free Multi-Scale Navigation in Virtual Environments” included in the papers section of the course notes.

References:

LaViola, J., Acevedo, D., Keefe, D., and Zeleznik R. “Hands-Free Multi-Scale Navigation in Virtual Environments”, In the Proceedings of the 2001 Symposium on Interactive 3D Graphics, 9-15, March 2001.

TULIP – Three Up Labels in Palm

- Menu system using Pinch gloves
- Derived from a number of iterations
- Non-dominant hand controls menus
- Dominant hand controls menu items



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

TULIP is an interaction tool which takes the concept of the 2D menus and brings it into a 3D interface. By utilizing Pinch gloves, the user has a menu system attached to the hands which is activated by pinching postures. The non-dominant hand holds menu choices and the dominant hand holds three menu items at a time which correspond to thumb to index, thumb to middle, and thumb to ring contacts. A “more” option is displayed on the pinkie which points to another set of three menu items shown in the palm of the hand. These three menu items will become available if the users makes a thumb to pinkie contact.

Although other 2D menu systems have been developed for 3D applications such as pull-down and body-centered menus, the TULIP system keeps the menu in the user’s hands rather than in the virtual space or by some other part of the body.

References:

Jacoby, R. and S. Ellis. “Using Virtual Menus in a Virtual Environment”, In SPIE: Virtual Data Interpretation, 1992.

Mine, M., F. Brooks, and C. Sequin. “Moving Objects in Space: Exploiting Proprioception in Virtual Environment Interaction”, ACM SIGGRAPH’97, 19-26, 1997.

TULIP – Evaluation

- Compared with pull-down and pen and “pen and tablet” menus
- “Pen and tablet” found to be faster
- Users preferred TULIP
- TULIP had higher comfort level

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

A user evaluation which compared TULIP with pull-down and “pen and tablet” menus was conducted to test TULIP’s ease of learning, efficiency and comfort. The results of the study indicate that although “pen and tablet” interaction was faster than TULIP, more users preferred the TULIP system and found it to be more comfortable to use than the other two menu techniques. More details on the user evaluation and the design of TULIP can be found in the paper, “Design and Evaluation of Menu Systems for Immersive Virtual Environments”, found in the papers section of the course notes.

References:

Bowman, D. and C. Wingrave, “Design and Evaluation of Menu Systems for Immersive Virtual Environments”, Proceedings of IEEE Virtual Reality 2001, 149-156, 2001.

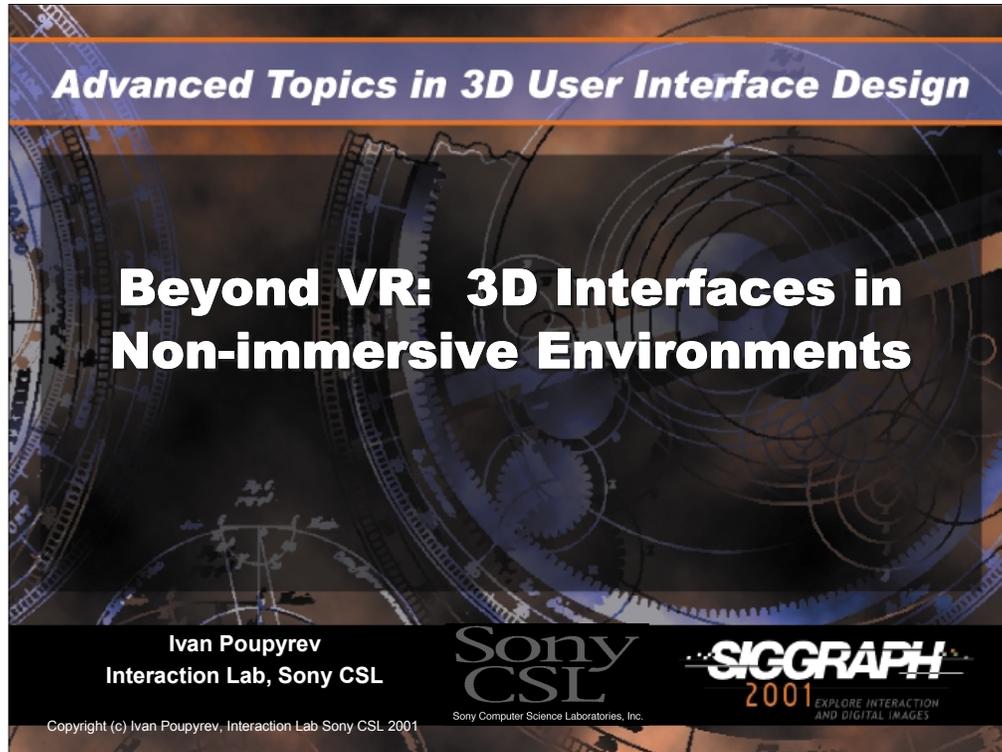
Lindeman, R., J. Silbert, and J. Hahn, “Hand-Held Windows: Towards Effective 2D Interaction in Immersive Virtual Environments”, Proceedings of IEEE Virtual Reality’99, 205-212, 1999.

Conclusions

- 2D interface metaphors can be critical in 3D applications
- Seamless integration of 2D and 3D components is essential
- Make the tools that the user needs intelligent
- Important to find lightweight solutions when using hand-held devices
- Field is still in its infancy

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

We must continue to explore how 2D interface concepts and components can fit into 3D interfaces and virtual environments. They are powerful tools when used properly and can greatly increase productivity for users.



Ivan Poupyrev, Ph.D.
Interaction Lab, Sony CSL

E-mail: poup@cs1.sony.co.jp

WWW: <http://www.csl.sony.co.jp/~poup/>

Address:
Interaction Lab, Sony CSL
Takanawa Muse Bldg.,
3 – 14 – 13 Higashigotanda
Shinagawa-ku, Tokyo 141-0022
Japan

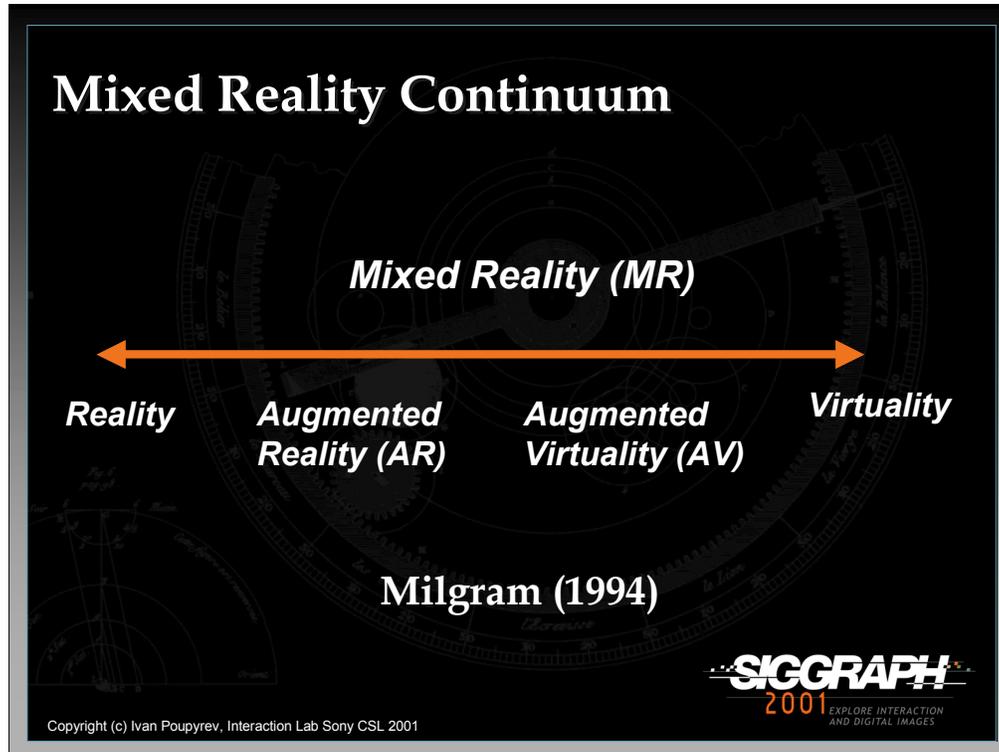
3D user interfaces outside computer

- **We live in 3D world**
 - seeing, hearing, touching and smelling are spatial skills
- **Computer-generated sensory stimulation already surrounds us - everyday part of the real world**
- **Augmenting physical space with interactive computer-controlled stimuli**
 - desktop 3D interfaces
 - augmented reality (AR)
 - ubiquitous computing
 - wearable computers
- **Designing 3D interface to the real world**

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

We live in a 3D world. Most of our natural sensory abilities, e.g. seeing, hearing, touching and even smelling, are spatial and allow to distinguish spatial positions, directions, shapes and forms of stimuli. By augmenting physical spaces with computing devices and computer controlled sensory stimuli we can create 3D user interfaces that are embedded into the physical world around us. The research areas that has been investigating these new interfaces is augmented and ubiquitous computing research and in some of the research work they attempt to design a 3D user interface to the real word.



The future living environments and even today's living spaces will probably represent continuum between the purely physical reality and pure virtuality, an approach which Milgram described as a Mixed Reality continuum (1994).

Lecture Overview

- **Real-world 3D user interfaces**
 - Desktop 3D user interfaces – do not cover
 - Augmented Reality and ubiquitous computing
 - Properties/challenges in mixed reality interface
- **Mixed reality interfaces**
 - Traditional approach: AR as information browser
 - Spatial, 3D AR interfaces
 - Augmented surfaces and tangible interfaces
 - Tangible AR interfaces
 - Agent based AR interfaces
- **Future research directions**

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

This slide outlines the contents of this lecture.

Desktop 3D interfaces

- **The most familiar 3D user interface**
 - 3D modeling applications
 - Computer games
 - Information visualization
 - 3D desktops for PC (Robertson, et al. 2000)
- **Adopting 2D devices, such as mouse to interact in 3D**



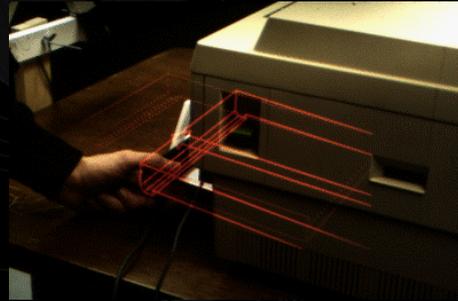
Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The most familiar 3D user interfaces are desktop 3D interfaces that has been widely used in 3D modeling, computer games, information visualization, etc. The major challenge in these interfaces is to adopt 2D devices such as mouse and keyboard to perform 3D tasks, e.g. navigation. There has been a lot of techniques implemented in the 2D interfaces, and these interfaces are not covered in this lecture.

Mixed Reality Interfaces

- **Following Azuma definition of AR (1997)**
 - a) combine real and virtual objects
 - b) interactive in real time
 - c) virtual objects are registered in 3D physical world



KARMA, Feiner, et al. 1993

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

This lecture will discuss mixed reality interfaces, e.g. interfaces that follow Azuma's definition: they superimpose virtual information on real world, they are interactive and spatial.



Therefore, I will not discuss interfaces such as Triangles (Gorbet, et al. 1998), which does in a sense combine virtual and real, but does not register virtual objects in 3D physical environment. Similarly, although large-scale projection screens are common in public spaces, and the virtual images that they display are sometimes registered to the surrounding environment, I would also not consider them as AR interfaces because they are not interactive.

Mixed and Augmented Reality Interfaces: Basic Technology

- **Tracking and registration**
 - reliable fast tracking of the user viewpoint in the
 - registration of virtual objects in physical world
 - 6DOF magnetic and computer vision tracking
- **Presentation and display technology**
 - See-through HMDs
 - Projection



Copyright University of North Carolina at Chapel Hill

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

The basic technologies required to built reliable AR systems are tracking and registration techniques as well as display technology to present the virtual image to the user. The most often used tracking techniques are magnetic, optical and computer vision tracking. The most often used display technologies are see-through HMDs and projection screens.

Challenges in AR Interfaces

- **Conflict between real world and virtual**
 - Not neatly separated anymore
- **Limitations of displays**
 - Precise, fast registration & tracking
 - Spatially seamless display
- **Limitations of controllers**
 - Precise, fast registration & tracking
 - Spatially seamless interactivity



Image Copyright Sony CSL

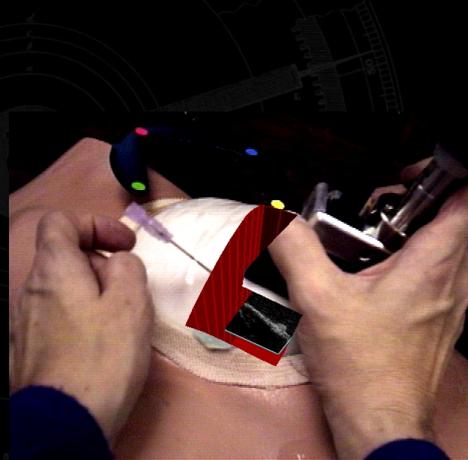
SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

AR has been traditionally used for visual augmentation, and it's only been relatively recently that there's growing interest in AR interaction issues. The most basic challenge in designing AR interfaces is a conflict between real and virtual: unlike in traditional VR the interfaces are not neatly separated into their own domains. More particularly, the design of AR interfaces is limited mostly by the properties and limitations of AR display technology and tracking and registration techniques. Optimally, the basic AR technologies should allow unobtrusive user interaction with virtual objects superimposed on 3D physical objects everywhere (hence the interface is everywhere). However, these technologies have their own particular properties and limitations, leading to very different interaction styles.

AR interfaces as 3D data browsers (I)

- **3D virtual objects are registered in 3D**
 - See-through HMDs, 6DOF optical, magnetic trackers
 - “VR in Real World”
- **Interaction**
 - 3D virtual viewpoint control
- **Applications**
 - Visualization, guidance, training



State, et al. 1996

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

The AR data browsing was one of the first applications of AR interfaces. They were in some sense designed to superimpose VR on the real world. Indeed, the main goal of these AR data browsers is to correctly register and render 3D virtual objects relative to their real world counterparts and user viewpoint position. For example, the medical field has used these techniques to support doctors decisions during medical procedures by superimposing real time physiological data on the patient (Bajura, 1993) and to guide doctors by displaying possible needle paths (State'96). Possible applications for aircraft wiring at Boeing and training applications (Feiner, 1993) have been also proposed. These AR systems are based on see-through HMDs and 6DOF optical and magnetic trackers. Interaction is usually limited to the real-time virtual viewpoint control to correctly display virtual objects.

AR interfaces as context based information browsers (II)

- Information is registered to real-world context
 - Hand held AR displays
 - Video-see-through (Rekimoto, 1997) or non-see through (Fitzmaurice, et al. 1993)
 - Magnetic trackers or computer vision based
- Interaction
 - Manipulation of a window into information space
- Applications
 - Context-aware information displays



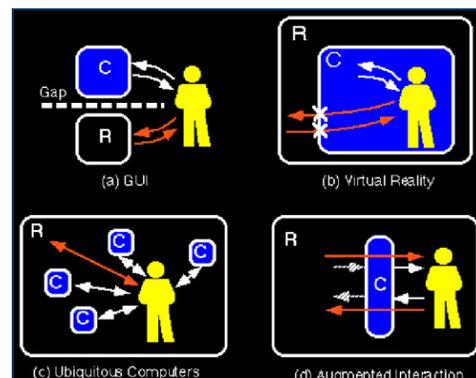
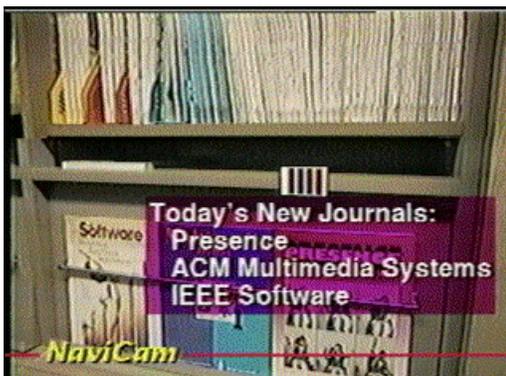
Rekimoto, et al. 1997



Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

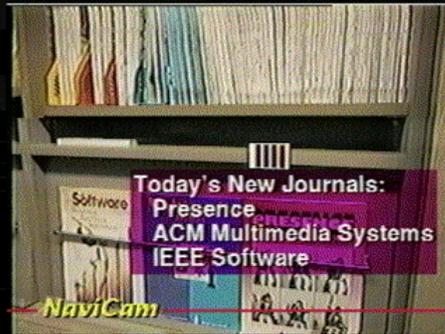
The data does not necessarily have to be 3D or modeled from the real world. Any information can be superimposed on the real world. Thus AR displays can present the data, e.g. text notes, voice or video annotations, etc, within a current real-world context. This approach was initially studied by Fitzmaurice (1993) in the Chameleon system and by Rekimoto (1997) in the NaviCam system. Hand-held displays were used to present information, using markers and a video see-through setup (Rekimoto, 1997) or magnetic trackers (Fitzmaurice, 1993). The interaction however was still limited to virtual viewpoint manipulation within the information space overlaid onto the physical world.

Rekimoto's NaviCam system and Augmented Interaction (1997)



AR Info Browsers (III): Pros and Cons

- **Important class of AR interfaces**
 - Wearable computers
 - AR simulation, training
- **Limited interactivity**
 - Modification and authoring virtual content is difficult



Rekimoto, et al. 1997

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

Viewing information superimposed on the physical world does not cover the spectrum of human activities. We also need to have an active impact on both the physical and virtual worlds, to actively change it. However, AR interfaces that act only as information browsers offer little opportunity to modify and author virtual information.

3D AR Interfaces (I)

- **Virtual objects are displayed in 3D space and can be also manipulated in 3D**
 - See-through HMDs and 6DOF head-tracking for AR display
 - 6DOF magnetic, ultrasonic, or other hand trackers for input
- **Interaction**
 - Viewpoint control
 - 3D user interface interaction: manipulation, selection, etc.



Kiyokawa, et al. 2000

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

The simplest and most natural approach to adding interactivity to information browsers is to use 6DOF input devices which are commonly used in VR interfaces, to allow the user to manipulate augmented virtual objects in 3D space. Virtual objects should still be presented in 3D using see-through head mounted displays, and magnetic or other tracking techniques. By interaction here I mean the traditional 3D interaction that is usually present in VR interfaces: 3D object manipulation, menu selection, etc. These features have been investigated by Kiyokawa et al. (2000) in SeamlessDesign, Ohshima et al. (1998) in AR2Hockey and Schmalsteig et al. (1996) in Studierstube, etc.

3D AR Interfaces (II): Information Displays

- How to move information in AR context dependent information browsers?
- InfoPoint (1999)
 - Hand-held device
 - Computer-vision 3D tracking
 - Moves augmented data between marked locations
 - HMD is not generally needed, but desired since there are little display capabilities



Khotake, et al. 1999

SIGGRAPH
2001 EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

InfoPoint (Khotake, 1999) adds 3D interaction to context-dependent information browsers, thereby providing the capability to move data within these environments. It's a hand-held device with a camera that can track markers attached to various locations in the physical environment, select information associated with the markers, and move it from one marker to another. InfoPoint does not require HMD, but because it has limited display capabilities, the feedback to the user is very limited.

3D AR Interfaces (III): Pros and Cons

- **Important class of AR interfaces**
 - Entertainment, design, training
- **Advantages**
 - Seamless spatial interaction: User can interact with 3D virtual object everywhere in physical space
 - Natural, familiar interfaces
- **Disadvantages**
 - Usually no tactile feedback and HMDs are often required
 - Interaction gap: user has to use different devices for virtual and physical objects

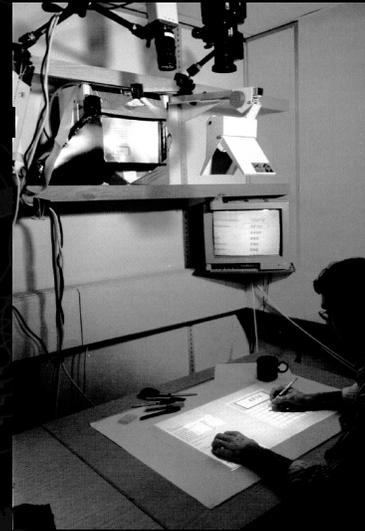
Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

SIGGRAPH
2001 EXPLORE INTERACTION
AND DIGITAL IMAGES

3D AR interfaces are important and have been used successfully in entertainment and design applications (e.g. Oshima, 2000). However, there is also insufficient tactile feedback, and HMDs are required. The user is also required to use different input modalities when handling physical and virtual objects: the user must use their hands for physical objects and special-purpose input devices for virtual objects. This introduces interaction seam into the natural flow of the interaction.

Tangible interfaces and augmented surfaces (I)

- **Basic principles**
 - Virtual objects are projected on a surface
 - back projection
 - overhead projection
 - Physical objects are used as controls for virtual objects
 - Tracked on the surface
 - Virtual objects are registered to the physical objects
 - Physical embodiment of the user interface elements
 - Collaborative



Digital Desk. 1993

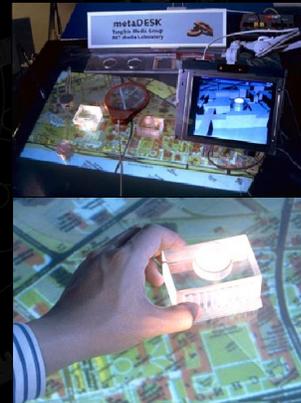
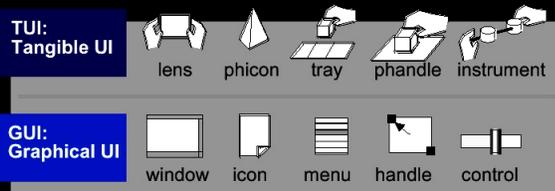
SIGGRAPH
2001 EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

The alternative approach to 3D AR is to register virtual objects on the surfaces, using either overhead or back projection. The user can then interact with virtual objects by using traditional tools, such as a pen, or specifically designed physical icons, e.g. phicons, which are tracked on the augmented surface using a variety of sensing techniques. This approach was first developed during the Digital Desk project (Wellner, et al. 1993) and has been further developed by other researchers such as Fitzmaurice, et al, 1995, Ullmer, et al. 1997, Rekimoto, 1998.

Tangible Interfaces and Augmented Surfaces (II)

- **Graspable interfaces, Bricks system (Fitzmaurice, et al. 1995) and Tangible interfaces, e.g. MetaDesk (Ullmer'97):**
 - Back-projection, infrared-illumination computer vision tracking
 - Physical semantics, tangible handles for virtual interface elements

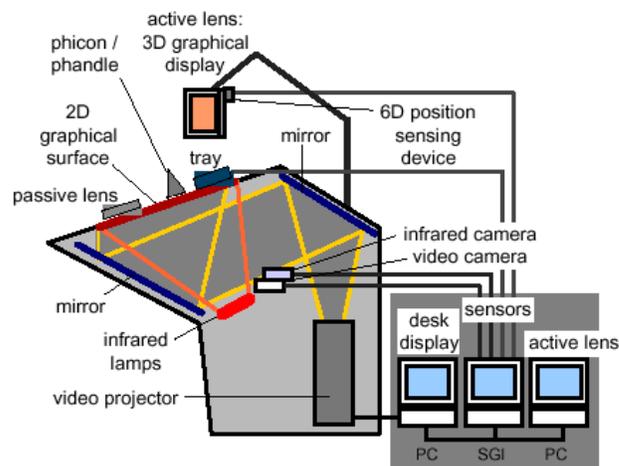


metaDesk. 1997



Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

An example of such a system is a metaDesk by Ullmer, et al. 1997. In this system, the image is back-projected on the table and the surface of the table is back-illuminated with infrared lamps. Physical objects on the table reflect the infrared lights and their position and orientation on the table surface can be tracked using an infrared camera located under the table (see figure below). Therefore, this system can track physical objects and tools and register virtual images relative to them, which allows us to manipulate and interact with the virtual images by using these physical, tangible handles. Different objects can be discerned on the table and used to control different interface functionality.

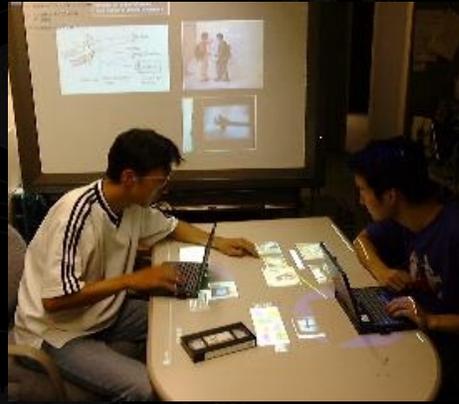


Configuration of the metaDesk (Ullmer, et al. 1997)

Tangible Interfaces and Augmented Surfaces (III)

- **Rekimoto, et al. 1998**

- Front projection
- Marker-based tracking
- Multiple projection surfaces
- Tangible, physical interfaces + AR interaction with computing devices



Augmented surfaces, 1998

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

Another approach is to use an overhead projection system such as in Rekimoto, et al. (1999) and Underkoffler, et al. (1998). Physical objects are tracked on the table by using markers attached to them. An overhead camera and computer-vision techniques enable us to estimate the objects' 2D positions on the table. The physical objects can then be used for interactions on the table, e.g. by manipulating them, we can select and move virtual objects. Rekimoto et al. (1999) further extended this, by linking multiple projection surfaces, and using traditional computer devices, for example laptop computers, to interact with virtual objects.

Tangible Interfaces and Augmented Surfaces (IV)

- **Advantages**
 - Seamless interaction flow – user hands are used for interacting with both virtual and physical objects.
 - No need for special purpose input devices
- **Disadvantages**
 - Interaction is limited only to 2D surface
 - Spatial gap in interaction - full 3D interaction and manipulation is difficult

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

In tangible interfaces and augmented surfaces, the same devices are used for interactions in both the physical and virtual world. I am talking here about human hand and traditional physical tools. Therefore, there is no need for special-purpose input devices, such as in case of 3D AR interfaces. The interaction, however, is limited to the 2D augmented surface. Full 3D interaction is possible, although difficult, and hence there is a spatial seam in the interaction flow.

Orthogonal nature of AR interfaces (Poupyrev, 2001)

	3D AR	Augmented surfaces
Spatial gap	No interaction is everywhere	Yes interaction is only on 2D surfaces
Interaction gap	Yes separate devices for physical and virtual objects	No same devices for physical and virtual objects

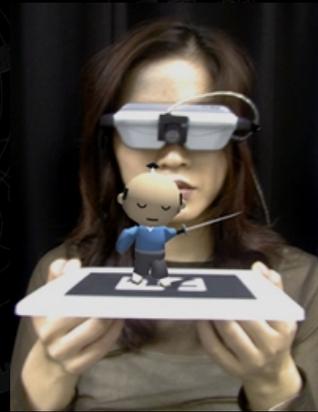
Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

It has been observed that the properties of 3D AR interfaces and augmented surfaces are somewhat orthogonal (Poupyrev, et al. 2000). 3D AR provides users with a spatially continuous environment, where 3D objects can be displayed and accessed from everywhere in space. At the same time, it introduces a seam into the interaction flow, requiring different devices for physical and virtual interactions. Augmented surfaces provide seamless interaction and the user can interact with virtual objects using physical tools or their hands. However, this does not allow for seamless spatial interaction, since the interaction is limited to the 2D space of the augmented surfaces.

Tangible AR interfaces (I)

- **Virtual objects are registered to marked physical “containers”**
 - HMD
 - Video-see-through tracking and registration using computer vision tracking
- **Virtual interaction by using 3D physical container**
 - Tangible, physical interaction
 - 3D spatial interaction
- **Collaborative**



Shared Space, 1999

SIGGRAPH
2001 EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

Using tangible augmented reality interfaces (Billinghurst, et al. 2000, Kato, et al. 2000, Poupyrev, et al. 2001) researchers are attempting to bridge the gap between 3D AR and augmented surfaces. Virtual objects are registered to marked physical objects in 3D using HMDs, video-see through AR registration techniques (using a camera mounted on the HMD), and computer-vision tracking algorithms. The user manipulates the virtual objects by physically manipulating the physical, tangible containers that hold them. Multiple users are able to interact with the virtual objects at the same time.

Tangible AR (II): generic interface semantics

- **Tiles semantics**

- data tiles
- operation tiles
 - menu
 - clipboard
 - trashcan
 - help

- **Operation on tiles**

- proximity
- spatial arrangements
- space-multiplexed



Tiles, 2001

SIGGRAPH
2001 EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

Tangible AR interfaces allow us to define generic interface elements and techniques, similar to GUI or tangible interfaces (Ullmer, 1997). This generic functionality has been investigated in the *Tiles* system (Poupyrev, et al. 2001). Tiles interface attempted to design a simple yet effective interface for authoring MR environments, based on a consistent interface model, by providing tools to add, remove, copy, duplicate and annotate virtual objects in MR environments.

The basic interface elements are *tiles* that act as generic tangible interface control, similar to icons in a GUI interface. Instead of interacting with digital data by manipulating it with a mouse, the user interacts with digital data by physically manipulating the corresponding tiles. There are three classes of tiles: *data tiles*, *operator tiles*, and *menu tiles*. All share a similar physical appearance and common operation. The only difference in their physical appearance is the icon identifying the tile type. This enables users who are not wearing an HMD to identify them correctly. *Data tiles* are generic data containers. The user can put and remove virtual objects from data tiles; if a data tile is empty, nothing is rendered on it.

-continued on the next page

Operator tiles are used to perform basic operations on data tiles, including *deleting* a virtual object from a data tile, *copying* a virtual object from a data tile to the clipboard or from the clipboard to a data tile, and requesting *help* and displaying annotations associated with a virtual object on the data tile. The operator tiles are identified by virtual 3D widgets attached to them.

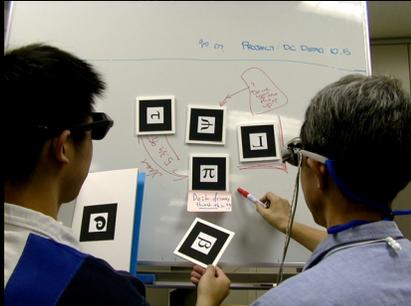
Menu tiles make up a book of the tiles attached to each page. This book works like a catalogue or a menu. As users flip through the pages, they can see the virtual objects attached to each page, choose the required instrument and then copy it from the book to any empty data tile.

Operations *between tiles* are invoked by putting two tiles next to each other (within a distance less than 15% of the tile size). For example, to copy an instrument to the data tile, users first find the desired virtual instrument in the menu book and then place an empty data tile next to the instrument. After a one-second delay to prevent accidental copying, a copy of the instrument smoothly slides from the menu page to the tile and is ready to be arranged on the whiteboard. Similarly, if users want to remove data from the tile, they put the trashcan tile close to the data tile, thereby removing the data from it.

Operation	Result
Menu operations	
 +  = 	
Clipboard operations	
 +  = 	
 +  = 	
 +  = 	
Trashcan operations	
 +  = 	
 +  = Not defined	
 +  = 	
Help operations	
 +  =  Message	
 +  =  Help	
 +  = Not defined	

Tiles semantics and operations on them (Poupyrev, et al. 2001)

Tangible AR (III): Space-multiplexed

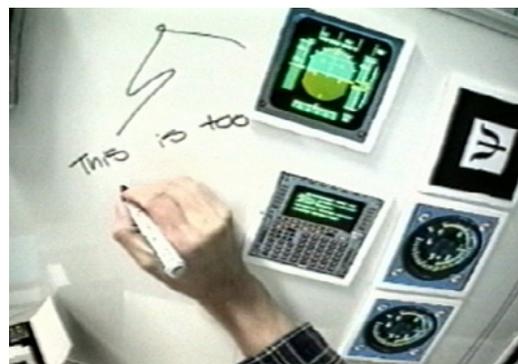


Data authoring in Tiles (Poupyrev, et al. 2001). Left, outside view of the system; right, view of the left participant.

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

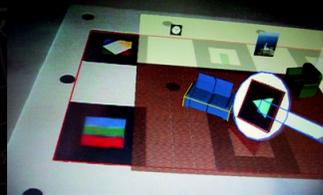
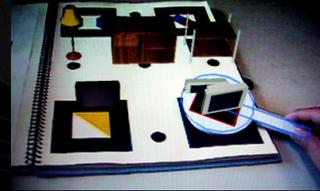
Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

Tangible AR environments provide an easy-to-use interface for the quick authoring of AR environments. For example, Poupyrev, et al. 2001, designed an interface for the rapid layout and prototyping of aircraft panels, thereby, allowing both virtual data and traditional tools, such as whiteboard markers, to be used within the same environment. This is an example of a space-multiplexed interface design using tangible augmented reality interfaces.



Annotating data in Tiles (Poupyrev, et al. 2001)

Tangible AR (IV): Time-multiplexed interaction



Data authoring in WOMAR interfaces (Kato et al. 2000). The user can pick, manipulate and arrange virtual furniture using a physical paddle.

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

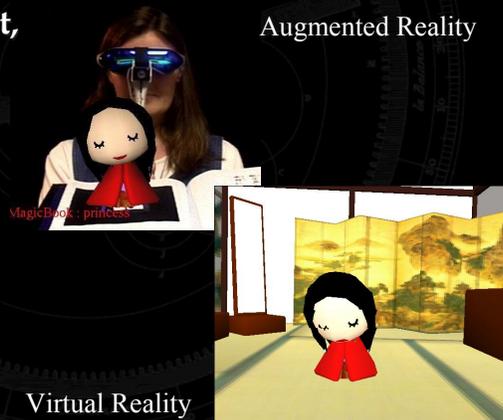
The VOMAR project (Kato, et al. 2000) explored how a time-multiplexed tangible AR interface could be designed. In the project, a single input device was used that allowed users to perform different tasks in a virtual-scene assembly application. The application was a layout of virtual furniture in a room, although the same interface could be applied to many domains. When users opened the book they saw a different set of virtual furniture on each of page, such as chairs, rugs etc. A large piece of paper on the table represented an empty virtual room. They could then copy and transfer objects from the book to the virtual room using a paddle, which was the main interaction device. The paddle is a simple object with an attached tracking symbol that can be used by either hand and enables users to use static and dynamic gestures to interact with the virtual objects. For example, to copy an object from the book onto the paddle users simply placed the paddle beside the desired object. The close proximity was detected, and the object was copied onto the paddle. The VOMAR system demonstrated how simple 6DOF interaction devices can be developed using the Tangible Augmented Reality approach.

Tangible AR (V): AR - VR Transitory Interfaces

- **Magic Book (Billinghurst, et al. 2001)**

- 3D pop-up book: a transitory interfaces

- Augmented Reality interface
- Portal to Virtual Reality
- Immersive virtual reality experience
- Collaborative



Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

The MagicBook project (Billinghurst, et al. 2001) explored how a tangible AR user interface can be used to smoothly transport users between reality and virtuality. The project did this by using a normal book as the main interface object. Users could turn the pages of the book, look at the pictures, and read the text without any additional technology. However, if they looked at the pages through an Augmented Reality display, they would see 3D virtual models appearing out of the pages. The AR view is, therefore, an enhanced version of a 3D “pop-up” book. Users could change the virtual models simply by turning the pages, and when they saw a scene they particularly liked, they could fly into the page and experience the story as an immersive virtual environment. In VR they were free to move about the scene at will and interact with the characters in the story or return back to the real world. The tangible user interface therefore provides a technique for the seamless blending of virtual reality experience to everyday user activities.

Tangible AR (V): Conclusions

• Advantages

- Seamless interaction with both virtual and physical tools
 - No need for special purpose input devices
- Seamless spatial interaction with virtual objects
 - 3D presentation of and manipulation with virtual objects anywhere in physical space

• Disadvantages

- Required HMD
- Markers should be visible for reliable tracking

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

There are several advantages of tangible AR interfaces. First, they are *transparent interfaces* that provide seamless two-handed 3D interaction with both virtual and physical objects. They do not require participants to use or wear any special purpose input devices or tools, such as magnetic 3D trackers, to interact with virtual objects. Instead users can manipulate virtual objects using the same input devices they use in the physical world – their own hands – which leads to seamless interaction between digital and physical worlds. This property also allows the user to easily use both digital and conventional tools in the same working space.

Tangible AR allows *seamless spatial interaction* with virtual objects anywhere in their physical workspace. The user is not confined to a certain workspace but can pick up and manipulate virtual data anywhere just, like real objects, and arrange them on any working surface, such as a table or whiteboard. The digital and physical workspaces are therefore continuous, naturally blending together.

AR Groove: Tangible AR without HMD

- AR Groove (Poupyrev, et al. 2000)
 - Overhead camera tracking
 - AR workspace on screen in front of the users
 - Spatial gestures for musical control
 - 3D AR widgets extend tangible controllers

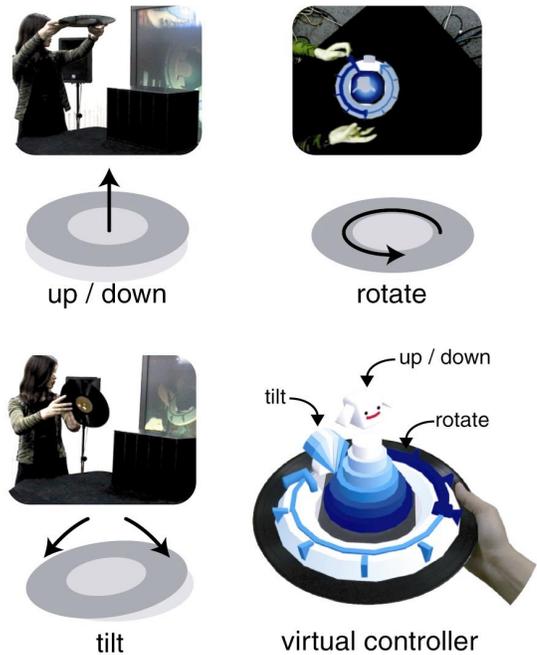


Augmented Groove, 2001



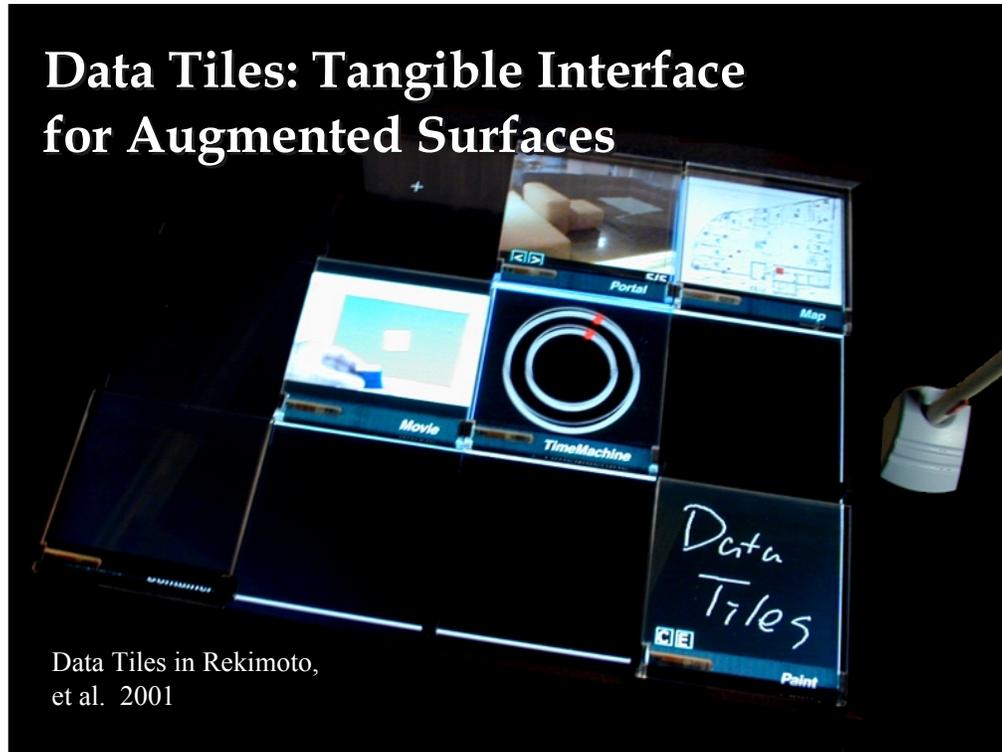
Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

AR Groove (Poupyrev et al., 2000) is a simple music controller for playing music that used tangible AR without HMDs. In AR Groove, the camera was installed on top of the table, and it tracked marked LP records. The performer controlled the music by manipulating vinyl LP records, and the user's spatial gestures, expressed through object manipulations, were mapped into musical modifications. Three simple gestures were used to control performance: *vertical translation*, *tilt*, and *rotation*. At the same time, the performer was presented with a simple visual display on the state of the controller, which provided immediate feedback on the process of performance. No HMDs, wires or special-purpose input devices were needed to play the music.



Gestures defined in AR Groove and virtual controller (Poupyrev, et al. 2000)

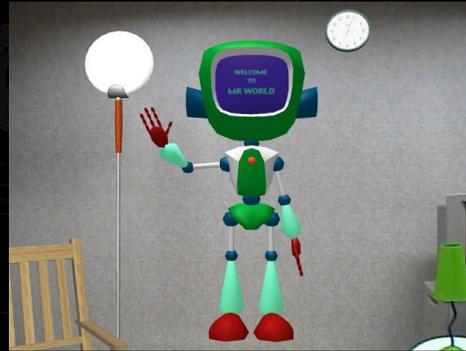
Data Tiles: Tangible Interface for Augmented Surfaces



An interesting approach related to tangible AR was also designed and investigated in the DataTiles system by Rekimoto, et al. 2001. In this system, the user could arrange and interact with the virtual data by using transparent tiles that were placed on a flat sensor-enhanced display, through which the image was presented to the user.

Agents in AR

- **Conversational AR agents:
Indirect interaction in AR**
 - ALIVE (Maes, et al. 1997)
 - Projection based, no HMD
 - Welbo (Anabuki, et al, 2000)
 - HMD-based
 - Speech and gesture interface
 - Embodiment, 3D interaction
- **Gesture and speech
recognition is still not
perfect**



Welbo AR agent,
copyright MR Lab, 2000

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

The final approach to designing AR interfaces is to use embodied agents, an approach which has been investigated in systems such as ALIVE (Maes, 1995) and Welbo (Anabuki, et al. 2000). The agent interface allows for gesture and speech command in AR environment. The user can ask agents to perform simple tasks such as moving furniture in the environment. The problem with these interfaces is that current techniques for gesture and speech recognition have not been perfected and some tasks cannot be effectively carried out by using verbal commands.

Wrap up

- **What have we learned?**
 - Why AR interfaces?
 - Traditional approach to AR interaction
 - 3D AR interfaces
 - Augmented surfaces and tangible AR interfaces
 - Orthogonality of 3D AR and AR surfaces
 - Tangible Augmented Reality interfaces
 - AR Agents-based interfaces
- **What is the future of AR interfaces?**

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

My talk has discussed some of the topics listed above.

Future research directions

- **Robotic AR interfaces**
- **Richer sensory displays**
 - Audio
 - Tactile
 - Smell and taste
- **Biometric controls**
 - Brain controls
 - Direct image transfer to the image centers
 - EMG controls, etc.



SIGGRAPH
2001 EXPLORE INTERACTION
AND DIGITAL IMAGES

Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

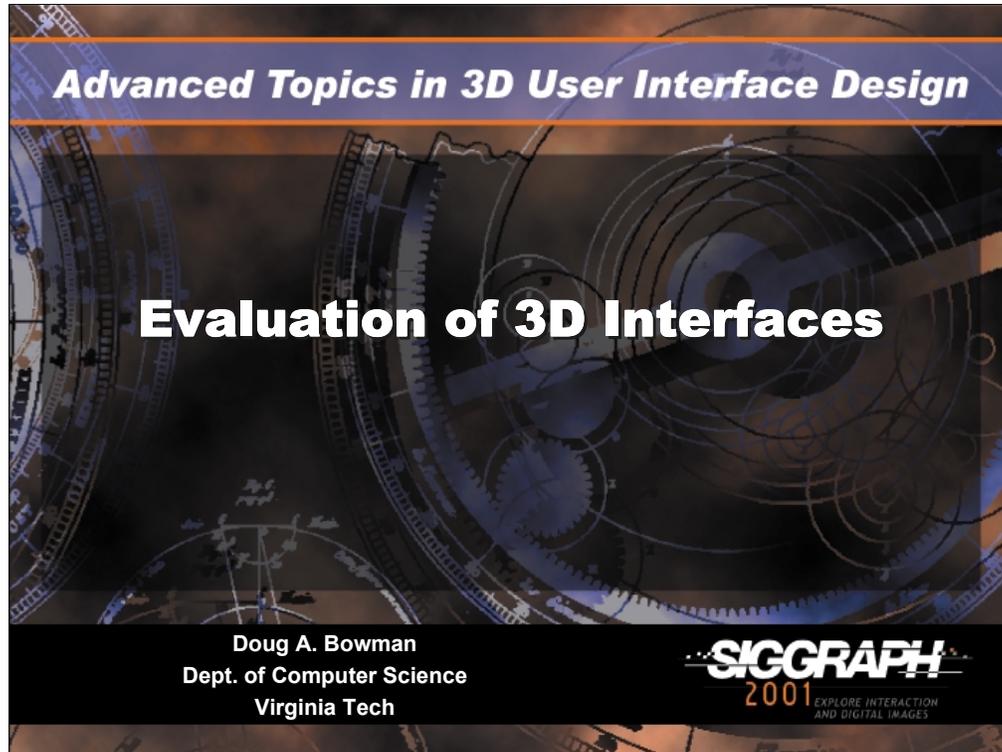
The future is exciting.

References

- Anabuki, M., Kakuta, H., Yamamoto, H., Tamura, H., Welbo: An Embodied Conversational Agent Living in Mixed Reality Spaces. *Proceedings of CHI'2000, Extended Abstracts*. 2000. ACM. pp. 10-11.
- Bajura, M., Fuchs, H., Ohbuchi, R., Merging Virtual Objects with the Real World: Seeing Ultrasound Imagery Within the Patient. *Proceedings of SIGGRAPH '92*. 1992. ACM. pp. 203-210.
- Billighurst, M., Poupyrev, I., Kato, H., May, R., Mixing Realities in Shared Space: An Augmented Reality Interface for Collaborative Computing. *Proceedings of ICME 2000*. 2000. IEEE. pp. 1641-1644.
- Billighurst, M., Kato, H., Poupyrev, I. The MagicBook: An Interface that Moves Seamlessly Between Reality and Virtuality. *IEEE Computer Graphics and Applications*, May/June 2001, pp. 2-4
- Feiner, S., MacIntyre, B., Seligmann, D., Knowledge-Based Augmented Reality. *Communications of the ACM*, 1993. 36(7): pp. 53-62.
- Fitzmaurice, G., Ishii, H., Buxton, W., Bricks: Laying the foundations for graspable user interfaces. *Proceedings of CHI'95*. 1995. ACM. pp. 442-449.
- Fitzmaurice, G.W., Situated information spaces and spatially aware palmtop computers. *Communication of the ACM*, 1993. 36(7): pp. 38-49.
- Gorbet, M., Orth, M., Ishii, H., Triangles: Tangible Interface for Manipulation and Exploration of Digital Information Topography. *Proceedings of CHI'98*. 1998. ACM.
- Ishii, H., Ullmer, B., Tangible bits towards seamless interfaces between people, bits and atoms. *Proceedings of CHI97*. 1997. ACM. pp. 234-241.
- Khotake, N., Rekimoto, J., Anzai, Y., InfoStick: an interaction device for Inter-Appliance Computing. *Proceedings of Handheld and Ubiquitous Computing*. 1999.
- Kato, H., Billighurst, M., Poupyrev, I., Imamoto, K., Tachibana, K., Virtual Object Manipulation on a Table-Top AR Environment. *Proceedings of International Symposium on Augmented Reality*. 2000.
- Kiyokawa, K., Takemura, H., Yokoya, N., SeamlessDesign for 3D Object Creation. *IEEE MultiMedia*, 2000. 7(1): pp. 22-33.
- Maes, P., The ALIVE system: wireless, full-body interaction with autonomous agents. *ACM Multimedia Systems*, 1997. 5(2): pp. 105-112.
- Milgram, P., Takemura, H., Utsumi, A., Kishino, F., Augmented Reality: A Class of Displays on the Reality-Virtuality Continuum. *Proceedings of Telem manipulator and Telepresence Technologies*. 1994. SPIE. pp. 282-292.
- Ohshima, T., Sato, K., Yamamoto, H., Tamura, H., AR2Hockey: A case study of collaborative augmented reality. *Proceedings of VRAIS'98*. 1998. IEEE. pp. 268-295.
- Ohshima, T., Satoh, K., Yamamoto, H., Tamura, H., RV-Border Guards: A Multi-Player Entertainment in Mixed Reality Space. *Proceedings of SIGGRAPH'2000 Conference Abstracts and Applications*. 2000. ACM. pp. 96.
- Poupyrev, I., Berry, R., Kurumisawa, J., Nakao, K., Billighurst, M., *et al.*, Augmented Groove: Collaborative Jamming in Augmented Reality. *Proceedings of SIGGRAPH'2000 Conference Abstracts and Applications*. 2000. ACM. pp. 77.
- Poupyrev, I., Tan, D., Billighurst, M., Kato, H., Regenbrecht, H., *et al.*, Tiles: A Mixed Reality Authoring Interface. *Proceedings of Interact 2001*. 2001.

-continued on the next page

- Rekimoto, J., Nagao, K., The World through the Computer: Computer Augmented Interaction with Real World Environments. *Proceedings of UIST'95*. 1995. ACM. pp. 29-36.
- Rekimoto, J., Saitoh, M., Augmented surfaces: A spatially continuous work space for hybrid computing environments. *Proceedings of CHI'99*. 1999. ACM. pp. 378-385.
- Schmalstieg, D., Fuhrmann, A., Szalavari, Z., Gervautz, M., Studierstube - An Environment for Collaboration in Augmented Reality. *Proceedings of CVE '96 Workshop*. 1996.
- State, A., Livingston, M., Hirota, G., Garrett, W., Whitton, M., *et al.*, Technologies for Augmented Reality Systems: Realizing Ultrasound-Guided Needle Biopsies. *Proceedings of SIGGRAPH'96*. 1996. ACM. pp. 439-446.
- Ullmer, B., Ishii, H., The metaDesk: Models and Prototypes for Tangible User Interfaces. *Proceedings of UIST'97*. 1997. ACM. pp. 223-232.
- Underkoffler, J., Ishii, H., Illuminating light: an optical design tool with a luminous-tangible interface. *Proceedings of CHI'98*. 1998. ACM. pp. 542-549.
- Wellner, P., Interaction with paper on the digital desk. *Communications of the ACM*, 1993. 36(7): pp. 87-96.



Evaluation of 3D interfaces

Doug A. Bowman
Dept. of Computer Science (0106)
660 McBryde Hall
Virginia Tech
Blacksburg, VA 24061 USA

Email: bowman@vt.edu

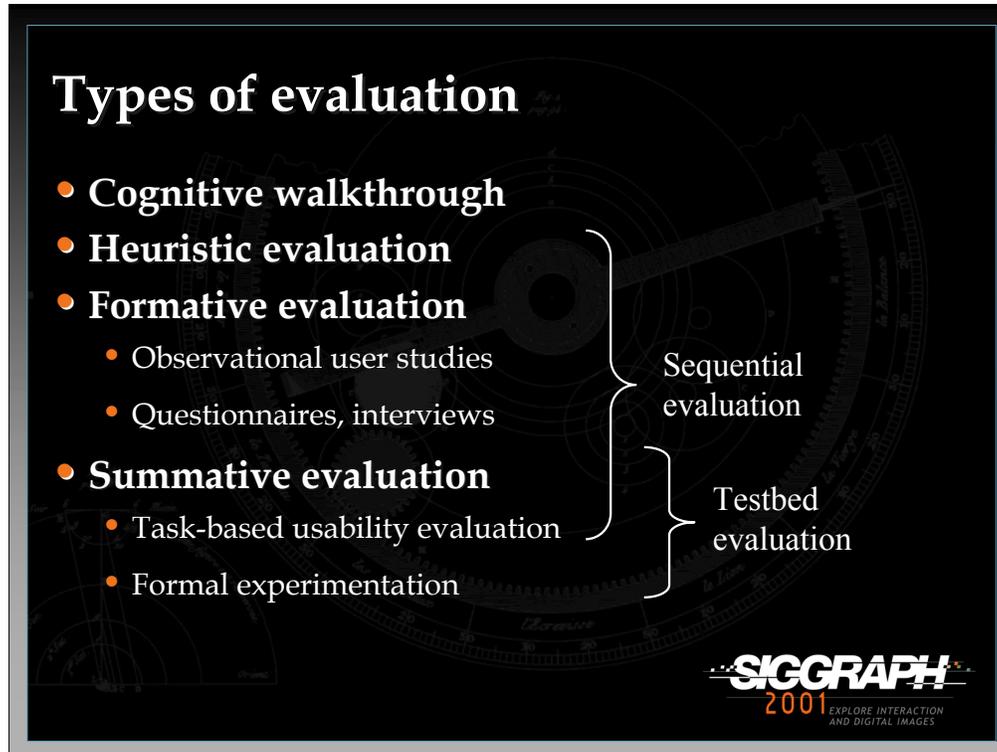
Web: <http://www.cs.vt.edu/~bowman/>

In this section, we'll discuss the evaluation of 3D interfaces and interaction techniques. Topics include:

- evaluation types
- evaluation issues
- how 3D UI evaluation differs from the evaluation of traditional interfaces
- evaluation approaches (testbed and sequential evaluation)
- metrics for 3D UI evaluation
- guidelines for 3D UI evaluation

-continued on the next page

We should note that systematic evaluation approaches are fairly new to 3D interface design. Until a few years ago, most researchers performed either cursory user studies or none at all, judging by published research papers. Thus, this is still an ongoing area of research. However, it draws heavily from traditional human-computer interaction (HCI) research. There are likely some readers who are not convinced of the necessity or usefulness of usability evaluation, but it's clear from experience and from the literature that a designer is not likely to produce a completely usable interface the first time – this is especially true for 3D interfaces, where there are fewer guidelines and examples from which to draw. Thus, assessment is necessary to catch the inevitable usability problems.



Here are some general categories of user interface evaluation that are applicable to 3D UIs.

A cognitive walkthrough is an evaluation done by experts, who step through each of the tasks in a system, asking detailed questions about each step in the task. For example, “Is it clear to the user what can be done here?”, or “Can the user translate his intention into an action?” The answers to these questions reveal potential usability problems.

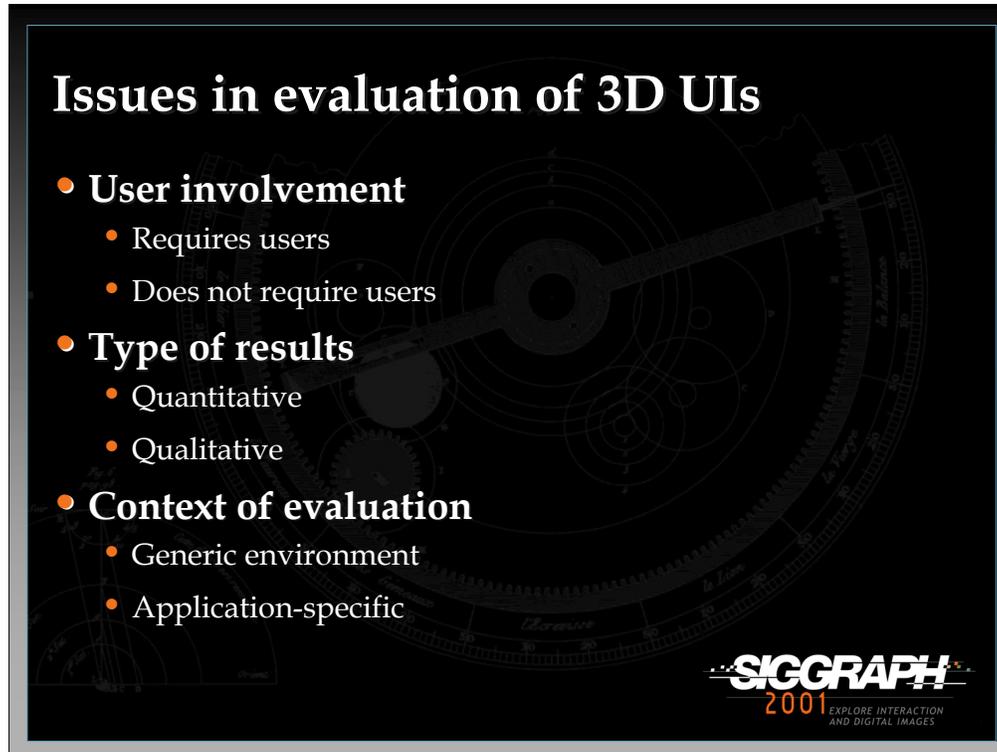
Heuristic evaluation refers to an evaluation by interface experts, using a well-defined set of heuristics or guidelines. Experts examine the interface visually, via a written description, or through actual use, and determine whether or not the interface meets the criteria set forth in the heuristics. For example, the interface might be checked to see if it meets the guideline: “Eliminate extraneous degrees of freedom for a manipulation task.”

Formative evaluations are used to refine the design of a widget, an interaction technique, or a UI metaphor. Observational user studies are informal sessions in which users try out the proposed interface. They may be asked to simply explore and play around, or to do some simple tasks. Often users’ comments are recorded (“think out loud” or verbal protocol), and the evaluator watches the user to see if there are parts of the interface that are frustrating or difficult. Post-hoc questionnaires and interviews may be used to get more detailed information from users about their experiences with the system.

Summative evaluations compare various techniques in a single experiment. A task-based usability evaluation is more structured. Users are given specific tasks to perform. Often, users are timed as they perform the tasks, and evaluators may keep track of errors made by the user. This information is then used to improve the interface. Formal experiments have a formal design including independent and dependent variables, subjects from a particular subject pool, a strict experimental procedure, etc. The results of formal experiments are usually quantitative, and are analyzed statistically.

We will be talking about two specific evaluation approaches in this section. Sequential evaluation spans a wide range of evaluation types. Testbed evaluation involves summative techniques.

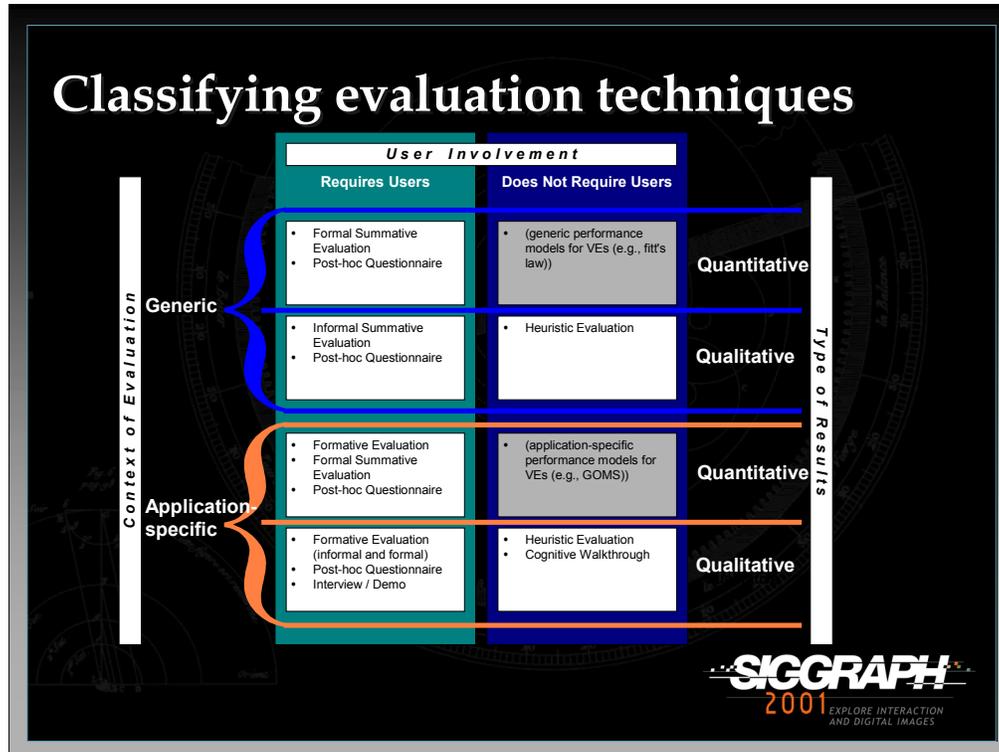
Note: Information in the first 12 slides is drawn from: Bowman, D., Gabbard, J., and Hix, D. Usability Evaluation in Virtual Environments: A Comparison and Integration of Methods. Submitted to the *ACM Transactions on Computer-Human Interaction*, 2001.



Evaluation methods can be classified according to three major issues:

1. **User involvement:** Some evaluation approaches use interface experts to make recommendations and suggestions for a system based on UI guidelines or principles. Other approaches utilize subjects drawn from the user population.
2. **Type of results:** Interface evaluations can produce quantitative (numeric) or qualitative (descriptive) results, or both. Both types can be extremely useful in refining or analyzing usability. Many approaches produce both types of results.
3. **Context of evaluation:** Some evaluations, especially formal experiments, are done in a generic testing environment, where the results can then be generalized to many applications. Other evaluations are application-specific, meaning that the results are more narrowly focused, but also that they may be more accurate.

Choosing an evaluation method can be seen as making a decision about each of these three issues. The choices you make depend on your specific goals and situation. For example, early in the design process, a qualitative evaluation not requiring users is cheap and fast, and will likely produce significant gains in usability. If I am designing general interaction techniques, a quantitative evaluation with users in a generic environment should produce general results regarding the situations for which the techniques are useful.



This slide shows various evaluation techniques classified according to the scheme from the previous slide.

The gray boxes represent parts of the design space that have not yet been explored in the context of the evaluation of 3D interfaces. We have suggested some possibilities for filling in these gaps. Both gaps have to do with the application of performance models for 3D interfaces. Such models do not yet exist due to the youth of the field.

How 3D UI evaluation is different

- **Physical issues**
 - User can't see physical world in HMD
 - Think-aloud and speech incompatible
- **Evaluator issues**
 - Evaluator can break presence
 - Multiple evaluators usually needed

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

There are a number of ways in which evaluation of 3D interfaces is different from traditional user interface evaluation.

First, there are physical issues. For example, in an HMD-based VE, the physical world is blocked from the user's view. This means that the evaluator must ensure that the user does not bump into objects or walls, that the cables stay untangled, and so on. Another example involves a common method in traditional evaluation called a "think-aloud protocol". This refers to a situation in which the user talks aloud about what he is thinking/doing in the interface. However, many 3D applications use speech input, which of course is incompatible with this evaluation method unless there is an explicit "push-to-talk" technique. Even in this case, the user could not invoke a command while simultaneously describing his thoughts/actions to the evaluator.

Second, we consider issues related to the evaluator. One of the most important is that an evaluator can break the user's sense of presence by talking to the user, touching the user, making changes to the environment, etc. during an evaluation. If the sense of presence is considered important to the task/application, the evaluator should try to avoid contact with the user during the tests. Another example of an evaluator issue is that multiple evaluators are usually needed. This is because 3D systems are so complex (hardware and software) and because users of 3D UIs have much more freedom and input bandwidth than users of a traditional UI.

How 3D UI evaluation is different (cont.)

- **User issues**
 - Very few expert users
 - Evaluations must include rest breaks to avoid possible sickness
- **Evaluation type issues**
 - Lack of heuristics/ guidelines
 - Choosing independent variables is difficult

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Third, we look at user issues. One problem is the lack of users who can truly be considered “experts” in 3D application usage. Since the distinction between expert and novice usage is important for interface design, this makes recruiting an appropriate subject pool difficult. Also, 3D systems have problems with simulator sickness, fatigue, etc. that are not found in traditional UIs. This means that the experimental design needs to include provisions like rest breaks and the amount of time spent in the system needs to be monitored.

Fourth, issues related to the type of evaluation performed. Heuristic evaluation can be problematic, because 3D interfaces are so new that there is not a large body of guidelines from which to draw, although this is changing. Also, if you are doing a formal experiment, there are a huge number of factors which might affect performance. For example, in a travel task, the size of the environment, the number of obstacles, the curviness of the path, the latency of the system, and the spatial ability of the user all might affect the time it takes a user to travel from one location to the other. Choosing the right variables to study is therefore a difficult problem. It’s also sometimes difficult to realize which factors must be held constant to avoid excessive variability in the experimental results.

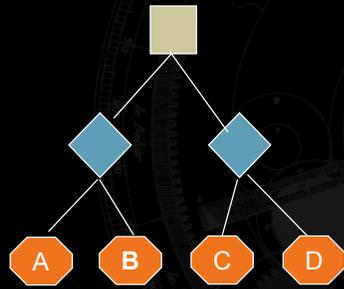
How 3D UI evaluation is different (cont.)

- **Miscellaneous issues**
 - Evaluations must focus on lower-level entities (ITs) because of lack of standards
 - Results difficult to generalize because of differences in 3D systems

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Finally, there are some miscellaneous issues related to 3D UI evaluation. Most interface evaluation focuses on subtle details of the interface, such as the placement of items within menus, or on the overall metaphor used in the interface. In 3D systems, however, evaluation often focuses on the basic interaction techniques, because we simply don't know yet what ITs should typically be used. Also, it's hard to generalize the results of an experiment or evaluation, because usually the evaluation is done with a particular type of hardware, a single type of environment, etc., but in real usage, a wide variety of different devices, software systems, and environments will likely be encountered.

Using taxonomies for evaluation



Techniques:
AC (15 secs.)
AD (10 secs.)
BC (20 secs.)
BD (? secs.)

- Taxonomy is a framework for evaluation
- Evaluation results at a more fine-grained level (evaluate technique components instead of complete techniques)
- Can lead to predictive power for performance of untested techniques

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

One way to impose some order on the huge design space for 3D interaction techniques is to create a classification or taxonomy of techniques. The type of taxonomy shown here is based on a hierarchic decomposition of a task. The *task* (tan box) is divided into *subtasks* that must be completed. For example, the task of object selection might be viewed as two subtasks: indicating an object, then giving a command to select that object. For each subtask, the taxonomy can list *technique components* that might be used to complete that subtask. In our example, pointing to the object is a component for the first subtask, and pressing a button is a component addressing the second subtask. A complete interaction technique is composed of a component for each lowest-level subtask. Therefore, in the figure, there are $2 \times 2 = 4$ possible interaction techniques that can be created.

Taxonomies can be used as a framework for evaluation. This not only means that you can design evaluations based on the structure given by a taxonomy, but that you can predict performance based on this structure as well.

-continued on the next page

Here's an example of predictive power:

Task: changing the color of an object

Subtasks: selecting object, selecting color

Technique components for object selection: pointing (A), choosing from list (B)

Technique components for color selection: RGB sliders (C), 3D RGB cube (D)

Technique 1: AC (measured to take an average of 15 seconds)

Technique 2: AD (10 seconds)

Technique 3: BC (20 seconds)

Technique 4: BD (not evaluated)

We can infer that component D takes 5 seconds shorter than C, and that B takes 5 seconds longer than A.

So BD can be calculated as $AD+5 = 15$ seconds, or $BC-5 = 15$ seconds.

Testbed evaluation framework

- **Main independent variables: ITs**
- **Other considerations (independent variables)**
 - task (e.g. target known vs. target unknown)
 - environment (e.g. number of obstacles)
 - system (e.g. use of collision detection)
 - user (e.g. VE experience)
- **Performance metrics (dependent variables)**
 - Speed, accuracy, user comfort, spatial awareness...
- **Generic evaluation context**



An evaluation testbed is a generalized environment in which many smaller experiments or one large experiment can be run, covering as much of the design space as you can. Like other formal experiments, you're evaluating interaction techniques (or components), but you also include other independent variables that could have an effect on performance. These include characteristics of the task, environment, system, and user.

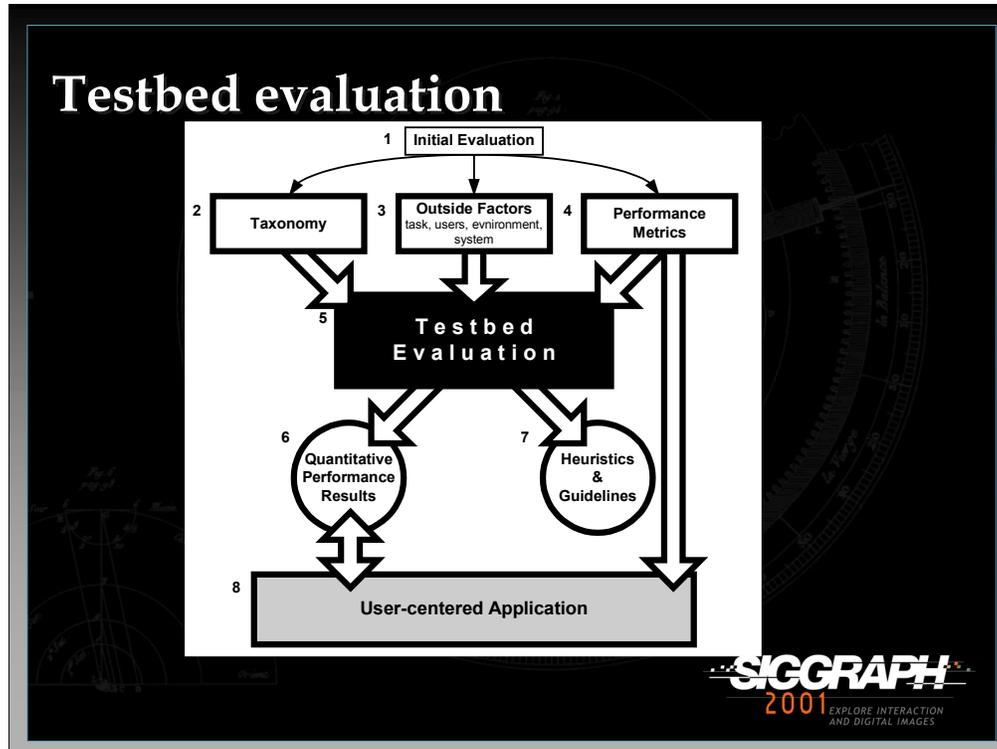
You also measure multiple dependent variables in such experiments to try to get a wide range of performance data. Here we use performance in the broader sense, not just meaning quantitative metrics. The more metrics you use, the more applications can use the results of the experiment by listing their requirements in terms of the metrics, then searching the results for technique(s) that meet those requirements.

Doug Bowman performed such evaluations in his doctoral dissertation, available online at: <http://www.cs.vt.edu/~bowman/thesis/>. A summary version of these experiments is in this paper:

Bowman, Johnson, & Hodges, Testbed Evaluation of VE Interaction Techniques, Proceedings of ACM VRST '99

Also see: Popyrev, Weghorst, Billingham, and Ichikawa, A Framework and Testbed for Studying Manipulation Techniques, Proceedings of ACM VRST '97.

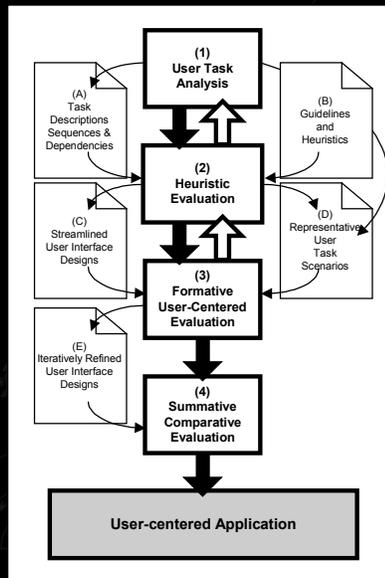
In terms of our three issues, testbed evaluation involves users, produces quantitative (and perhaps qualitative) results, and is done in a generic context.



This figure shows the process used in testbed evaluation. Before designing a testbed, one must understand thoroughly the task(s) involved and the space of interaction techniques for those tasks. This understanding can come from experience, but it's more likely to come from some initial (usually informal) evaluations. This can lead to a taxonomy for a task, a set of other factors that are hypothesized to affect performance on that task, and a set of metrics (discussed later).

These things are then used to design and implement a testbed experiment or set of experiments. The results of running the testbed are the actual quantitative results, plus a set of guidelines for the usage of the tested techniques. The results can be used many times to design usable applications, based on the performance requirements of the application specified in terms of the performance metrics.

Sequential evaluation



- Traditional usability engineering methods
- Iterative design/eval.
- Relies on scenarios, guidelines
- Application-centric

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

A different approach is called sequential evaluation. See the paper:

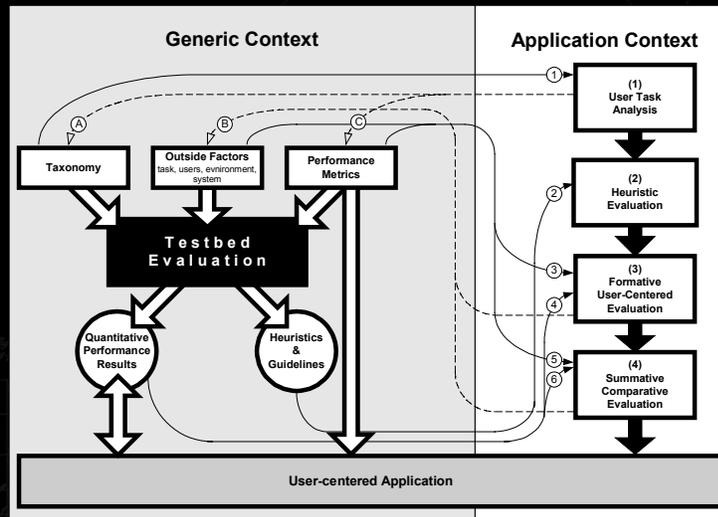
Gabbard, J. L., Hix, D, and Swan, E. J. (1999). User Centered Design and Evaluation of Virtual Environments , *IEEE Computer Graphics and Applications*, 19(6), 51-59.

As the name implies, this is actually a set of evaluation techniques run in sequence. The techniques include user task analysis, heuristic evaluation, formative evaluation, and summative evaluation. As the figure shows, the first three steps can also involve iteration. Note that just as in testbed evaluation, the goal is a user-centered application.

In terms of our three issues, sequential evaluation uses both experts and users, produces both qualitative and quantitative results, and is application-centric.

Note that neither of these evaluation approaches is limited to being used for the evaluation of 3D UIs. However, they do recognize that applications with 3D UIs require a more rigorous evaluation process than traditional 2D UIs, which can often be based solely on UI guidelines.

Combined approach



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

We have proposed to combine and integrate the testbed and sequential evaluation approaches. Recall that the major difference between the two is the context of evaluation: a generic context for testbed evaluation and an application context for sequential evaluation. This allows the results of the testbed process to be used as inputs for the sequential process, and vice-versa.

[letters and numbers refer to the arrows in the figure]

Using testbed evaluation as an input to sequential evaluation:

User task analysis, a critical part of the sequential evaluation approach, requires an understanding of tasks users must perform and possible ITs that could be used to accomplish those tasks. Taxonomic structures from the testbed approach provide both of these (1). Taxonomies provide a standard way to organize and decompose a task, and they contain a design space from which many ITs can be built.

The general guidelines produced by testbed evaluation can serve as input for heuristic evaluation in the sequential evaluation approach (2). In fact, this addresses a potential problem with using heuristic evaluation for VEs: a lack of heuristics. Since guidelines from the testbed approach are based on experimental evidence, heuristic evaluation using these guidelines should produce a more usable initial design to be fed to the formative evaluation process.

-continued on the next page

The set of factors other than ITs that could influence performance (outside factors) are an important component of the testbed evaluation process, since they are candidates for independent variables in testbed experiments. For example, one could test whether the number of obstacles in an environment affects the speed of traversing a path in that environment. These same factors can play a role in shaping formative and summative evaluation components of the sequential evaluation approach (3 and 5). The evaluator can use these factors to more carefully plan task scenarios that assess the range of potential interactions a user could have with the VE. In a similar way, sets of performance metrics defined for testbed evaluation are useful in formative and summative evaluation. These metrics can be checked to ensure that the evaluator observes all variables that contribute to a usable interface.

Finally, quantitative performance results obtained from testbed experiments can play a role in the sequential evaluation process. In formative evaluation (4), an evaluator is trying to produce one or more usable ITs that can later be compared. If testbed results are available for the task in question, incorporation of these ITs into a VE can begin at a much more refined level based on performance results. In the same way, testbed results can help narrow the set of ITs in summative evaluation (6). The relative performance of two ITs may already be known through testbed evaluation, or a particular IT may be known to perform badly in the situation presented by a particular VE application. In any case, these results should be considered before beginning either type of evaluation.

Using sequential evaluation as an input to testbed evaluation:

In all three of these cases, the experiences of analyzing a real-world application help to refine the generic model used for testbed evaluation.

One way this can occur involves the process of user task analysis (A). Task analysis takes place in the context of a particular application, and can also be refined as the sequential evaluation approach is iterated. This can result in a quite detailed understanding of user tasks, intentions, and mental models for a specific VE. This understanding is exactly what is needed to create good taxonomies of ITs for a particular task, since taxonomies in the testbed approach are based on task decomposition. If taxonomies more closely fit the user's model of a particular task, when this taxonomy is used as a framework for evaluation the results obtained should be a better predictor of user performance in real systems.

-continued on the next page

Subsequent to the process of user task analysis, usability goals and associated metrics can be determined. It is important for a user to complete tasks efficiently, correctly, without frustration, and in comfort. These characteristics match some of the possible performance metrics given by the testbed approach. However, it is possible that in the process of user task analysis and subsequent setting of usability goals, evaluators will find that a VE has a requirement whose fulfillment cannot be determined using any of the listed performance measures (C). The requirement may suggest a new metric to be added to the list and included in future testbed experiments.

It is difficult in the testbed approach to come up with complete lists of the outside factors that could affect performance. This is often done based on intuition alone. However, experiences of evaluators performing formative and summative evaluations can add to and refine these lists (B). Evaluators may notice that a user performing a particular task is greatly affected by some characteristic of the environment. This would suggest that this characteristic should be studied in a future testbed experiment to determine the extent of its effects more generally. If that variable has already been studied in a general experiment, it may be possible to give more weight to this factor in analysis of the results.

When is a 3D UI effective?

- Users' goals are realized
- User tasks done better, easier, or faster
- Users are not frustrated
- Users are not uncomfortable

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Now we turn to metrics. That is, how do we measure the characteristics of a 3D UI when evaluating it? I will focus on the general metric of *effectiveness*. A 3D UI is effective when the user can reach her goals, when the important tasks can be done better, easier, or faster than with another system, and when users are not frustrated or uncomfortable. Note that all of these have to do with the *user*. As we will see later, typical performance metrics like speed of computation are really not important in and of themselves. After all, the point of the application is to serve the needs of the user, so speed of computation is only important insofar as it affects the user's experience or tasks.

How can we measure effectiveness?

- **System performance metrics**
 - Avg. frame rate (fps), avg. latency / lag (msec) variability in frame rate / lag, network delay, distortion
- **Interface performance / User preference metrics**
 - Ease of learning, ease of use, presence, comfort
- **User (task) performance metrics**
 - Speed, accuracy, domain-specific metrics (learning, expressiveness, spatial orientation)
- **All are interrelated**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

We will talk about three different types of metrics, all of which are interrelated.

System performance refers to traditional CS performance metrics, such as frame rate.

As mentioned earlier, the only reason we're interested in system performance is that it has an effect on interface performance and user performance. For example, the frame rate probably needs to be at "real-time" levels before a user will feel present. Also, in a collaborative setting, task performance will likely be negatively affected if there is too much network delay.

Interface performance (the user's preference or perception of the interface) refers to traditional HCI metrics like ease of learning.

These metrics are mostly subjective, and are measured via qualitative instruments, although they can sometimes be quantified. For VE systems in particular, presence and user comfort can be important metrics that are not usually considered in traditional UI evaluation.

-continued on the next page

High levels of the user preference metrics generally lead to *usability*. A usable application is one whose interface does not pose any significant barriers to task completion. Often HCI experts will speak of a “transparent” interface – a UI that simply disappears until it feels to the user as if he is working directly on the problem rather than indirectly through an interface. User interfaces should be intuitive, provide good affordances (indications of their use and how they are to be used), provide good feedback, not be obtrusive, and so on. An application cannot be effective unless it is usable (and this is precisely the problem with some more advanced VE applications – they provide functionality for the user to do a task, but a lack of usability keeps them from being used).

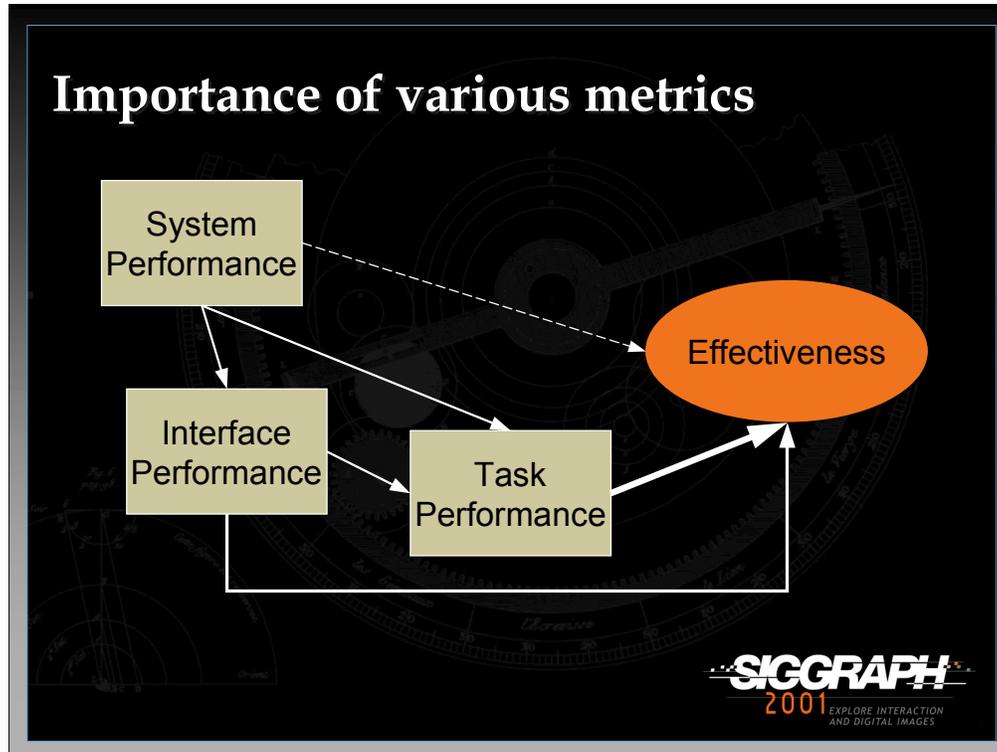
Presence is a crucial, but not very well-understood metric for VE systems. It is the feeling of being there – existing in the virtual world rather than in the physical world. How can we measure presence? A simple measure simply asks users to rate their feeling of “being there” on a 1-100 scale. Questionnaires generally ask many questions, all designed to get at different aspects of presence. Psychophysical measures are used in controlled experiments where stimuli are manipulated and then correlated to user’s ratings of presence (for example, how does the rating change when the environment is presented in mono vs. stereo modes?). There are also some more objective measures. Some are physiological (how the body responds to the VE). Others might look at users’ reactions to events in the VE (e.g. does the user duck when he’s about to hit a virtual beam). Tests of memory for the environment and the objects within it might give an indirect measurement of the level of presence. Finally, if we know a task for which presence is required, we can measure users’ performance on that task and infer the level of presence. Witmer and Singer (*Presence* 7(3), 1998) developed a formal presence questionnaire (PQ), along with an immersive tendencies questionnaire (ITQ). They did a lot of work to validate the design of these questionnaires, and have used them extensively to provide a standard measurement for presence. However, there is still a lot of controversy about their approach.

-continued on the next page

The other novel user preference metric for 3D systems is user comfort. This includes several different things. The most notable and well-studied is so-called “simulator sickness” (because it was first noted in things like flight simulator). This is similar to motion sickness, and may result from mismatches in sensory information (e.g. your eyes tell your brain that you are moving, but your vestibular system tells your brain that you are not moving). There is also work on the physical aftereffects of being exposed to 3D systems. For example, if a VE mis-registers the virtual hand and the real hand (they’re not at the same physical location), the user may have trouble doing precise manipulation in the real world after exposure to the virtual world. More seriously, things like driving or walking may be impaired after extremely long exposures (1 hour or more). Finally, there are simple strains on arms/hands/eyes from the use of 3D hardware. User comfort is also usually measured subjectively, using rating scales or questionnaires. The most famous questionnaire is the simulator sickness questionnaire (SSQ) developed by Robert Kennedy (*International Journal of Aviation Psychology*, 3(3), 1993). Kay Stanney has attempted some objective measures in her study of aftereffects – for example by measuring the accuracy of a manipulation task in the real world after exposure to a virtual world.

User (task) performance refers to the quality of performance of specific tasks in the 3D application, such as the time to complete a task.

The problem with measuring speed and accuracy is that there is an implicit relationship between them: I can go faster but be less accurate, or I can increase my accuracy by decreasing my speed. It is assumed that for every task there is some curve representing this speed/accuracy tradeoff, and users must decide where on the curve they want to be (even if they don’t do this consciously). So, if I simply tell my subjects to do a task as quickly and precisely as possible, they will probably end up all over the curve, giving me data with a high level of variability. Therefore, it is very important that you instruct users in a very specific way if you want them to be at one end of the curve or the other. Another way to manage the tradeoff is to tell users to do the task as quickly as possible one time, as accurately as possible the second time, and to balance speed and accuracy the third time. This gives you information about the tradeoff curve for the particular task you’re looking at.



This is an imprecise diagram that shows how I relate the three types of metrics. System performance directly affects interface performance and task performance. It only indirectly affects overall effectiveness of the 3D application (it's possible for the system to perform at low levels but still be effective). Interface performance and usability affect task performance directly, and also affects overall effectiveness directly, since an unusable 3D application will not be tolerated by users. Task performance is the most important factor in determining overall effectiveness, since the goal of the 3D UI is to allow users to do their tasks better, easier, and faster.

Guidelines for 3D UI evaluation

- **Begin with informal evaluation**
- **Acknowledge and plan for the differences between traditional UI and 3D UI evaluation**
- **Choose an evaluation approach that meets your requirements**
- **Use a wide range of metrics – not just speed of task completion**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Here are a set of guidelines to be used in any type of evaluation of 3D UIs.

Informal evaluation is very important, both in the process of developing an application and in doing basic interaction research. In the context of an application, informal evaluation can quickly narrow the design space and point out major flaws in the design. In basic research, informal evaluation helps you understand the task and the techniques on an intuitive level before moving on to more formal classifications and experiments.

Remember the unique characteristics of 3D UI evaluation from the beginning of this talk when planning your studies.

There is no optimal evaluation technique. Study the classification presented in this talk and choose a technique that fits your situation.

Remember that speed and accuracy do not equal usability. Also remember to look at learning, comfort, presence, etc.

Guidelines for formal experiments

- **Design experiments with general applicability**
 - Generic tasks
 - Generic performance metrics
 - Easy mappings to applications
- **Use pilot studies to determine which variables should be tested in the main experiment**
- **Look for interactions between variables – rarely will a single technique be the best in all situations**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

These guidelines are for formal experiments in particular – mostly of interest to researchers in the field.

If you're going to do formal experiments, you want the results to be as general as possible. Thus, you have to think hard about how to design tasks which are generic, performance measures that real applications can relate to, and a method for applications to easily re-use the results.

In doing formal experiments, especially testbed evaluations, you often have too many variables to actually test without an infinite supply of time and subjects. Small pilot studies can show trends that may allow you to remove certain variables, because they do not appear to affect the task you're doing.

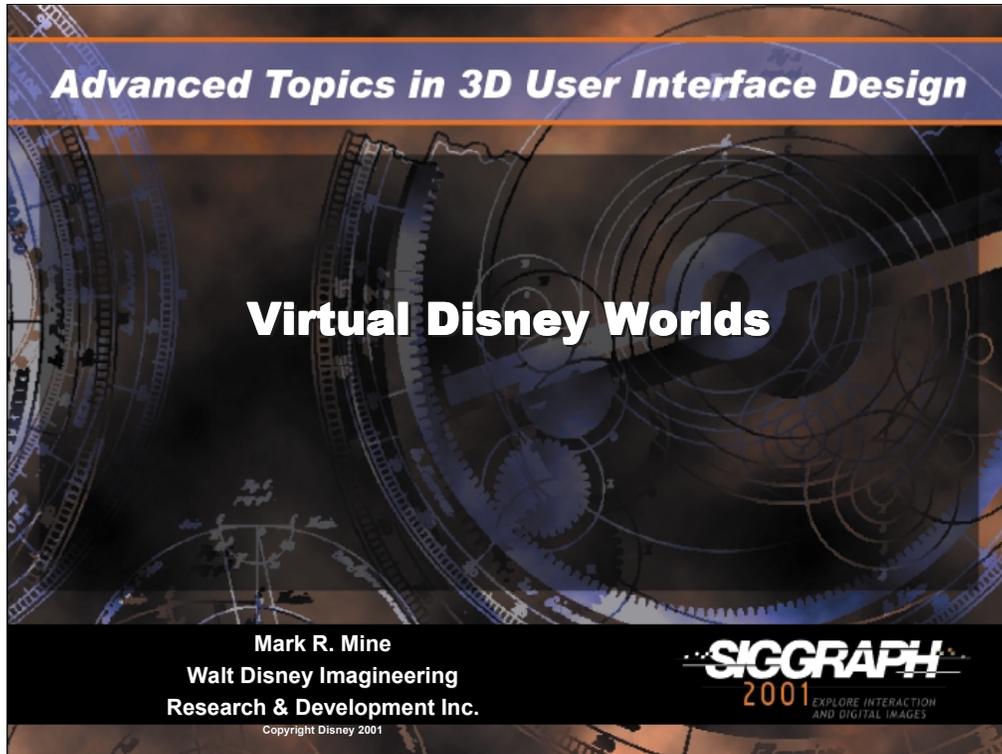
In almost all of the experiments we've done, the most interesting results have been interactions. That is, it's rarely the case that technique A is always better than technique B. Rather, technique A works well when the environment has characteristic X, and technique B works well when the environment has characteristic Y. Statistical analysis should reveal these interactions between variables.

Acknowledgments

- Deborah Hix
- Joseph Gabbard

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

I worked closely with Joe Gabbard and Debby Hix (both of Virginia Tech) in developing the list of distinctive characteristics of 3D UI evaluation, the classification scheme for evaluation techniques, and the combined testbed/sequential evaluation approach.



Outline

- VR Studio Background/Overview
- Projects Overview
- VR For Location Based Entertainment
- Working with Designers, Case Studies
- Lessons Learned

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Today what I'd like to do is give you an overview of the virtual worlds research at Disney

I'm going to begin by giving you an overview of the VR Studio.

Next I'll present some brief highlights of some of the projects we've been involved in the past, and talk about some of the things that we are currently working on.

Then I will discuss our experiences developing attractions for the Location Base Entertainment venue DisneyQuest

The remainder of the talk will focus on some of the work we've been doing in the area of simulation and visualization for theme park design. I'll try to relay to you some of the lessons we've learned working with designers

VR Studio - Then

- **Established in 1992 to explore the potential of VR technology for theme park attractions.**
 - Aladdin's Magic Carpet Ride

Mission Statement:
Advance the frontier of visual
quality and interactivity in
computer graphics for the Walt
Disney Company



- The VR Studio is part of Walt Disney Imagineering's (WDI) Research and Development Department
- WDI was traditionally responsible for the construction of Disney Theme parks. R&D was organized to develop new technologies to support that. Several years ago R&D's charter was expanded to encompass research and development for the entire company, and in particular help out with some of the new business units such as ABC.
- Originally we began developing an HMD/VR experience based upon the movie Rocketeer, but then changed focus to the movie Aladdin.
- Multiple versions of Aladdin
 - Aladdin Mark 1: single person, single scene
 - Aladdin Mark 2: single person, multiple scenes
 - Aladdin Mark 3: multi-player, multiple scenes

VR Studio - Now

- Location Based Entertainment
- 3D/4D Visualization for Theme Parks
- Interactive Experiences for the home
- High-quality pre-rendered graphics and animation for TV and theme park attractions

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Now we have expanded our portfolio considerably to encompass many different aspects of interactive computer graphics



- DisneyQuest is an indoor interactive park that combines the magic of Disney with cutting-edge immersive technologies, such as virtual reality and real-time 3-D. Spanning five floors, DisneyQuest has four unique zones of entertainment bursting with attractions, rides, and games. DisneyQuest locations are open at the Walt Disney World resort in Orlando and in downtown Chicago at Ohio and Rush Streets (description from www.disneyquest.com)

- DisneyQuest boasts multiple high-end attractions emphasizing real-time, interactive, computer graphics

- Cyberspace Mountain – Design your own roller coaster and ride it in a 2 axis, 360 degrees continuous rotation motion platform.
- Invasion! An Alien Encounter – 4 Player, pod based shoot-em-up utilizing infinity optics
- Ride the Comix – HMD based sword battle with cartoon characters
- Virtual Jungle Cruise – River raft ride on WDI-R&D designed air cell motion platform
- Aladdin’s Magic Carpet Ride – VR Studio’s HMD based multi-player ride through the movie Aladdin
- Hercules in the Underworld – VR Studio’s CAVE based attraction based on the movie Hercules – 4 players battle Hades in the underworld
- Pirates of the Caribbean – CAVE based attraction developed by the VR Studio. Motion platform, cannons, pirates, sea serpents and more.

3D Interface Design for LBE

- **Highly constrained by the unique nature of an LBE attraction**
 - Short 4 – 5 minute experience
 - Must be enjoyable by people 8 – 80
 - Operational considerations preclude the use of complex devices
- **Interaction primarily limited to navigation**
- **Usability key – focus on natural skills**

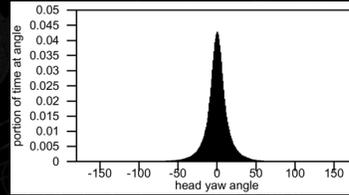
SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES



Screen shot from Aladdin's Magic Carpet ride. Entrance into the Sultan's palace.

Aladdin Interface Challenges

- **Intuitive interface for 3D navigation**
 - How do you fly a magic carpet?
- **Directing guest attention**
 - Complicated by limited HMD FOV
- **Encouraging interaction between guests in a shared virtual space**
 - Shared audio key



SIGGRAPH
2001 EXPLORE INTERACTION
AND DIGITAL IMAGES

Graph is from: *Disney's Aladdin: First Steps Toward Storytelling in Virtual Reality*, Randy Pausch, Jon Snoddy, Robert Taylor, Scott Watson, Eric Haseltine, **ACM SIGGRAPH 96 Conference Proceedings**, August 1996



Control seat and HMD used in Aladdin's Magic Carpet ride.



Control seat and HMD used in Aladdin's Magic Carpet ride.



Close up of “GatorVision” HMD developed by R&D for Aladdin’s magic carpet ride (subsequently marketed by nVision). High-res, CRT based design. Note cables attached to the front of the HMD used for weight relief. HMD also included an adjustable/detachable head unit which was given to the user before they entered the ride. Optics and earphones then quickly snapped onto head unit. This greatly improved ride load/unload times and made it possible for the head unit to be cleaned between users.



Screen shot from the CAVE based attraction, Hercules in the Underworld. 4 guests control four avatars (Hercules, Megara, Pegasus, and Phil) using 2 axis joysticks.

Hercules Interface Challenges

- **Intuitive device for controlling 3D Avatar**
 - Most 3D devices too complex for 5 minute experience
- **Shared viewpoint complicates camera control**
 - 4 users, single viewpoint, no head-tracking

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Interfaces for LBE attractions face the arcade challenge...arcade games typically have room for only three instructions, and by default the first two are:

- 1) Insert coin
- 2) Press start

Interfaces must therefore be immediately intuitive. The simple joystick is hard to beat.

Several techniques were used to augment Hercules camera control:

- Layout of the world was designed to prevent guests from wandering too far apart.
- A “bubble” surrounding guests was used to keep the camera centered on the avatars.
- A “pusher” mechanism was implemented to gently nudge guests along the story path if they stayed in one area too long.
- Story points were added that could be used to transport guest instantly to new scenes.



Interior shot of the prototype Hercules CAVE, 5 rear-projection screens (5 sides of a hexagon – only 3 visible in this shot). Projectors were turned on their side for greater vertical aspect ratio. Joysticks are custom designed with 1 inch thick steel shafts down the middle to withstand the rigors of daily use.



IMAX active stereo glasses used in the Hercules attraction (same glasses were used for the Pirates of the Caribbean ride). Blinders at the side of the glasses were removed to maximize peripheral vision (so guests could see screens to the side and behind them – though not in stereo they would at least provide motion cues).



Screen shot from Pirates of the Caribbean – Battle for Buccaneer Gold – the latest CAVE based attraction developed by the VR Studio (winner of the 2001 THEA award from the Themed Entertainment Association)



Images from the original Pirates of the Caribbean attraction at Disneyland.



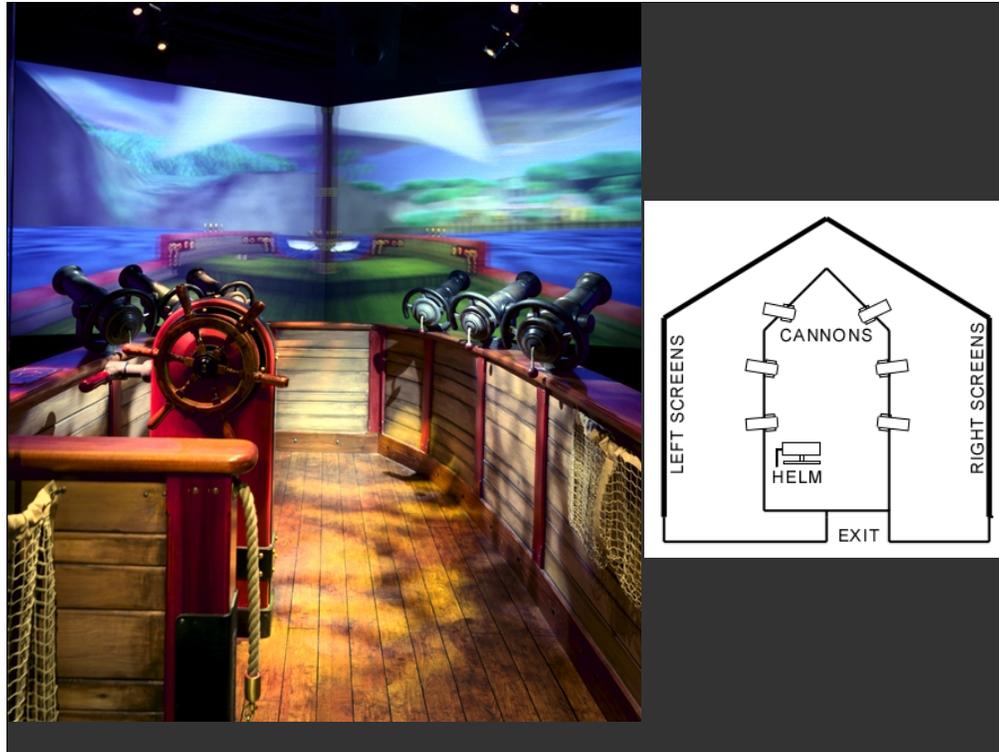
Extensive research was conducted in the original Pirates attraction. Here the authors interviews one of the animatronic figures in the ride.



Early storyboard sketch of the virtual pirates attraction.



Photo illustration of the final product (actual display screens not shown)

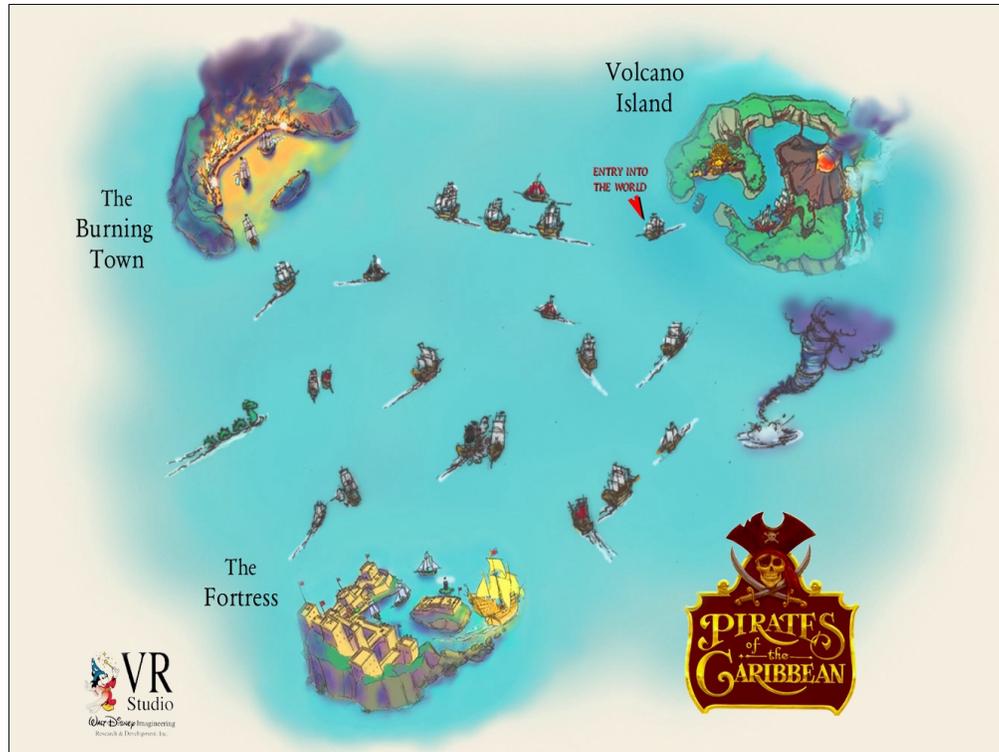


Screen shot and schematic diagram of a single Pirate's CAVE. Four rear-projection stereo displays. Air cell motion platform in the center with 6 physical cannons and a single helm. With one guest steering at a real helm, the other three guests man six real cannons to defeat virtual enemy pirate ships, forts, sea monsters and ghostly skeletons to collect and defend as much gold as possible in the five minute experience. *Pirates* uses wrap-around 3D screens, 3D surround sound, and a motion platform boat to fully engage the guest as a pirate.

Note: images are doubled on the screen due to stereo display.



Exterior of one of the Pirate's CAVEs. In order to ensure the high throughput that theme parks demand, there must be no time wasted acclimating the guest to the story, interface, or game rules. One thing *Pirates* makes extensive use of is an incredibly rich back-story that every guest can relate to – that of being a pirate. The attraction title, music, and theming of the queue line immediately gets the guest in the correct mind-set to play. They know what to expect, what is expected of them, and can then focus on the details of the interface and game rules.



Overview of the game world in the virtual pirates ride. A version of this map can be seen outside the entrance to the pirates CAVE in the previous slide. Since guests are free to roam throughout the world, exploring islands and battling ships, its important to acclimate them to the environment before they begin playing the game.



Pirates make extensive use of soft-skinned animated characters. Here Jolly Roger the Ghost Pirate explains the roles of the captain and gunners, and encourages the players to sink many pirate ships in order to get their gold.

Lessons Learned

*The importance of physical interfaces...
Especially for facile camera control*

- Aladdin's flying carpet interface
- Pirate's steering wheel
- Pirate's cannon

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Working With Designers

- **Goal: Virtual simulation and visualization for theme park design**
- **Challenge: Incorporating new tools into existing design process**
- **Competition: Proud tradition of physical model building**
 - High level-of-detail works of art
 - Simultaneous, low-latency, perspective correct viewing by unlimited viewers

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

At this point, what I'd like to do is focus in on some of the work we've been doing in the area of simulation and visualization for theme park design.

The goal was to apply our experience in the design of virtual spaces for location-based entertainment to the design of physical spaces for theme parks.

It was clear to us in the studio that we could provide the theme park designers some effective tools for visualization of future rides and attractions. Our challenge was to incorporate these tools into the existing design process.

What we were competing with is a 45 year tradition of building scale models for the visualization of these rides. To call these things models is to sell them short, these are incredible, high level-of-detail works of art. Furthermore, these models have many desirable characteristics which are hard to match in our current virtual systems.



Photograph of a physical scale model built for the Indiana Jones Ride at Disneyland.



Photograph of a different view of the same model.



Photograph of the same scene in the actual ride.



Photograph of the physical scale model built for the Pooh's Hunny Hunt attraction at Tokyo Disneyland.



Close up of the model showing the incredible level of detail in the model.

The Case for VR

Virtual simulations offer several key advantages over existing techniques:

- Rapid modifications to existing models
- Interactive sight-line evaluation
- Macro and micro scales in same model
- Visualization of complex behavior
 - Wave effect for Paradise Pier
 - Tigger Bounce for Pooh's Hunny Hunt

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

What key advantages do we have to offer the designers with our virtual systems?

The Disney Advantage

Why VR works at Disney:

- **Large-scale, high-cost construction projects benefit greatly from VR**
 - Unique designs
 - Customized materials
 - Specialized construction techniques
- **In-house artistic talent helps maximize effectiveness of our VR visualizations**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

A keen understanding of lighting and form enables talented Disney artist to create highly effective models for virtual simulations.

Case Study: Paradise Pier

- One of three major sections of Disney's California Adventure
- Initially hired to visualize coaster launch wave effect
- The power of 3D visualization obvious early on
 - Design flaws identified and corrected early in the design cycle

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Designers quickly realized that virtual walkthroughs made it easy for them to spot and correct sightline problems early on (avoiding costly corrections during the construction process).



Screen shot of the virtual simulation done by the VR Studio of the Paradise Pier section of Disney's California Adventure. Large building in the background is the Paradise Pier hotel which is across the street from the park.



A scale model built during the design of Paradise Pier.



Physical model of the California Screamin' roller coaster. Though it conveys a good understanding of the form of the ride, designers can not get a sense of the ride experience from the model.

Paradise Pier Visualization

- **Interactive 3D model enables multiple forms of visualization:**
 - Designer walkthroughs
 - Ride simulations
 - Sightline analysis
 - 4D simulations (3D model + time) for construction planning/ visualization

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

We amortized the cost of building the 3D model by reusing it in multiple simulations:

- Lagoon show design/development
- Crowd flow simulation/analysis
- Rescue/safety simulation/analysis



Exterior view of the R&D CAVE. 5 sided design similar to that used in the Hercules attraction.

Paradise Pier Video

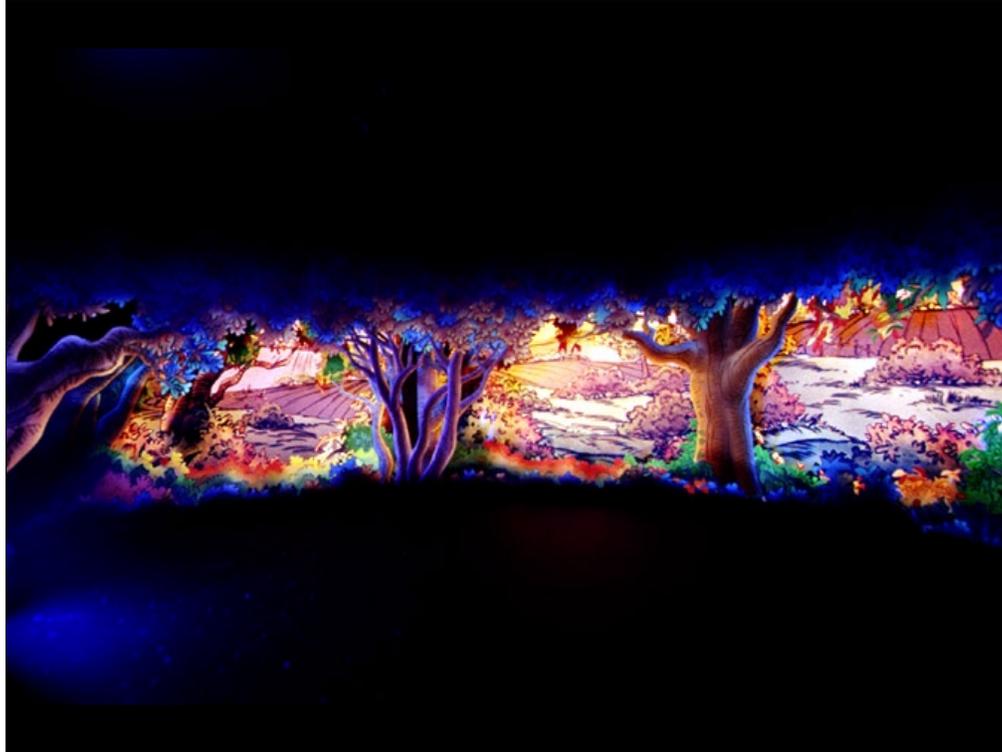
Paradise Pier Flythrough
4D Simulation
California Screamin' Simulation

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Case Study: Pooh's Hunny Hunt

- Major attraction developed for Tokyo Disneyland
- Originally hired to visualize Tigger bounce effect. Can we make the guests feel like they're bouncing with Tigger?
 - 1 bouncing car
 - 2 layers bouncing scenery
 - 4 layers bouncing video

SIGGRAPH
2001 EXPLORE INTERACTION
AND DIGITAL IMAGES



Black-lit physical model of the interior of the Tigger bounce section of Pooh's Hunny Hunt.



Virtual simulation of the Tigger bounce. Virtual cars moved up and down. Eye point could be moved to any seat in any car.

Pooh's Hunny Hunt

- **Simulation effort quickly expanded to include verification of ride timings**
 - Free-ranging computer controlled vehicles
 - Too complex for miniature cameras/models or pre-rendered visualizations
- **VR simulation enabled designers to quickly evaluate ride profiles from the guest's perspective**
 - 2D ride planning tool misleading

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

3D Hunny Hunt Model also utilized in multiple ways:

- Sightline verification used to validate reduction in number of audio-animatronic figures
- Real-time model used in planning of media development
 - Audio timing
 - Tigger bounce visuals



Overhead view of the Blustery Day and Tigger bounce sections of the Pooh's Hunny Hunt virtual model. This is a view of the VR studio's 3D simulation of the ride. The actual ride planning tool was much more iconic and represented the ride as a 2 dimensional image (plan view) with simple circles representing the ride vehicles.



Owl's house in the Hunny Hunt physical scale model.



Owl's house in the VR simulation.

Pooh's Hunny Hunt Video

Free-Ranging Vehicle
And Tigger Bounce Simulation

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Lessons Learned

The importance of CAVEs as a display medium

- **Large number of simultaneous viewers encourages interactive design sessions**
 - Demos into Design Sessions
- **Powerful communication tool**
 - Paradise Pier pre-bid
 - Selling Hunny Hunt to Oriental Land Company
- **The importance of first person perspective for ride timing verification**
- **Externalizes discussions**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Future Directions

DIRECT: Disney's Interactive Real-time Environment Construction Tools



DIRECT

- **Hardware**

- Rear-projection desktop with pen-based input
- Large FOV stereo projection screen for immersive viewing
- 6 DoF tracking for head-tracked stereo and direct manipulation
- Flexible device layer for incorporating joysticks, buttons, and other physical controls

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

DIRECT

- **Features**

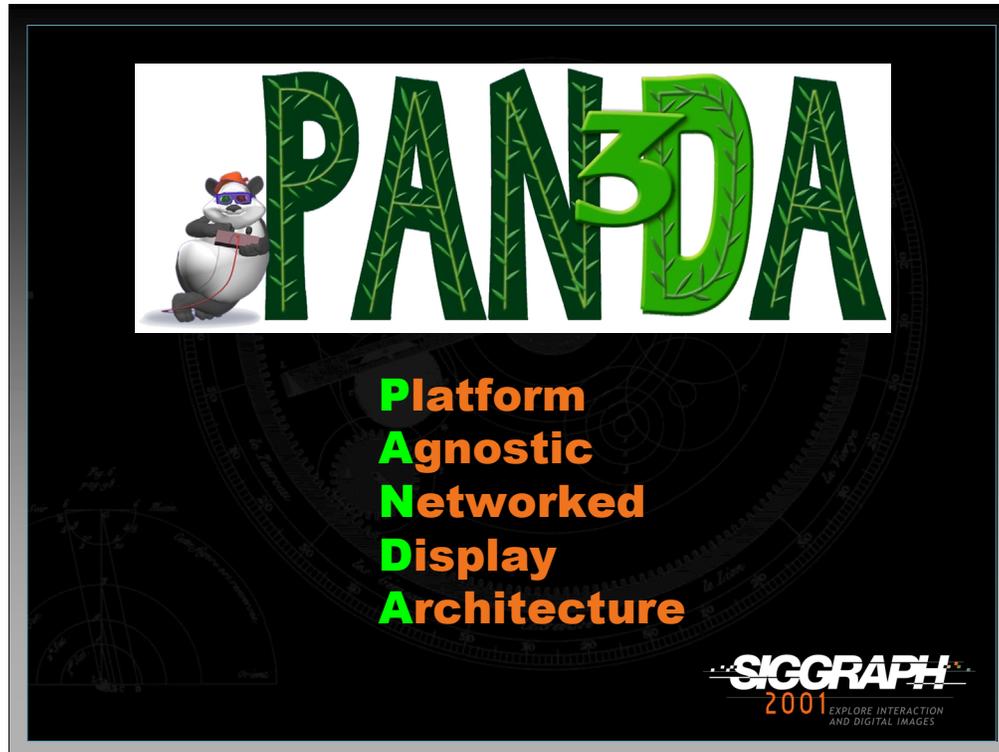
- Intuitive direct manipulation interface for placing and sizing of 3D objects
- Late-binding scripting layer for flexible control of dynamic simulations
- Powerful tools for 3D curve editing and camera control

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Lessons Learned

- **The importance of in-the-world tools**
 - Object placement/control tools
 - Curve editing for camera/object paths
 - Animation controls
- **Need to better span the space of display devices!**
 - Tools which work from desktop to CAVE

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES



DIRECT is built on top of PANDA3D, a fourth generation **open-source** VR software system developed by the VR Studio. PANDA 3D:

- Provides real-time 3D rendering
- Incorporates powerful tools for rapid prototyping based on DIRECT and Python
- Is platform agnostic (will run on multiple platforms.... Windows, Linux, IRIX, etc....)

Visit <http://www.panda3d.org> for more details

Lessons Learned

- **The importance of late-binding languages**
 - Interactive scripting layer (based upon Scheme/Squeak/Python) on top of high-performance C++ layer
 - Rapid implementation/iteration of dynamic environments
 - On-the-fly GUI building critical for flexible simulation control
- **Impossible to predict what designer needs**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES



The true challenge in VR research!



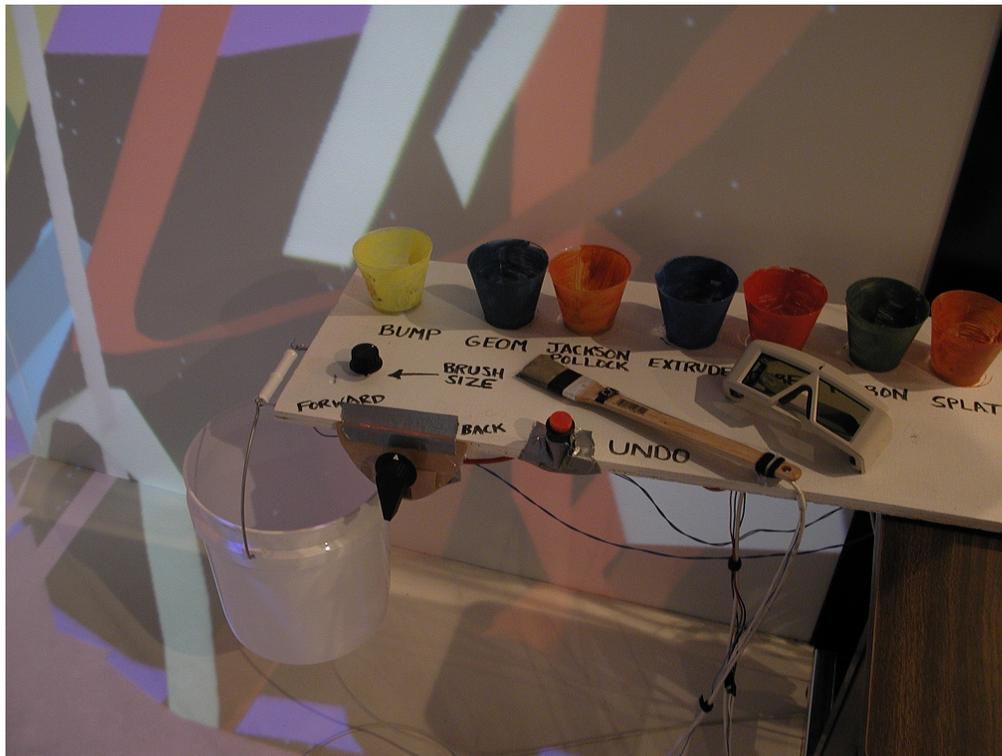
Flex and Pinch Input

The Flex and Pinch input system prototype. Although a CyberGlove is shown, any bend-sensing glove can be used.



ErgoDesk

A user creates 3D geometry at the ActiveDesk with a pen and performs camera operations using the 3D tracker in his non-dominant hand.



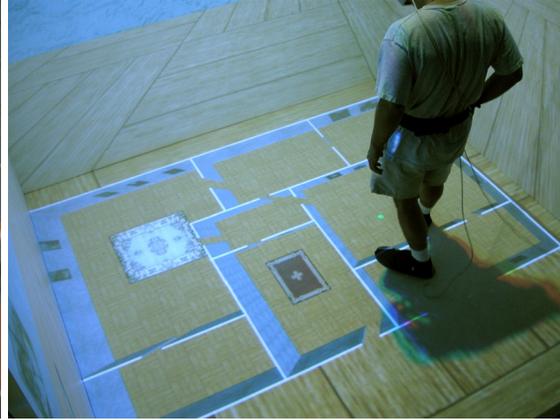
The Painting Table

An interface table made up of physical props used in the CavePainting application. Users can dip the brush into the paint cups to change the brush stroke style. The paint bucket is used to dump paint onto the virtual canvas.



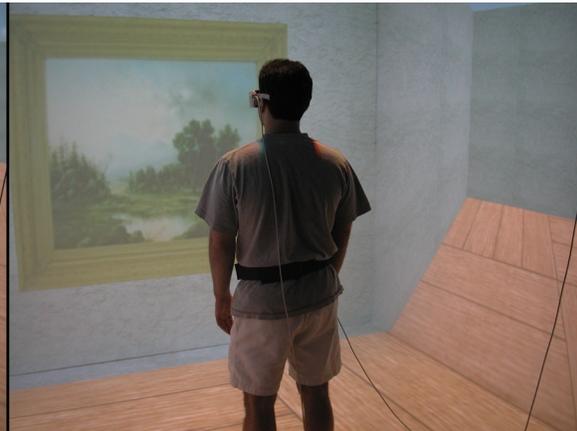
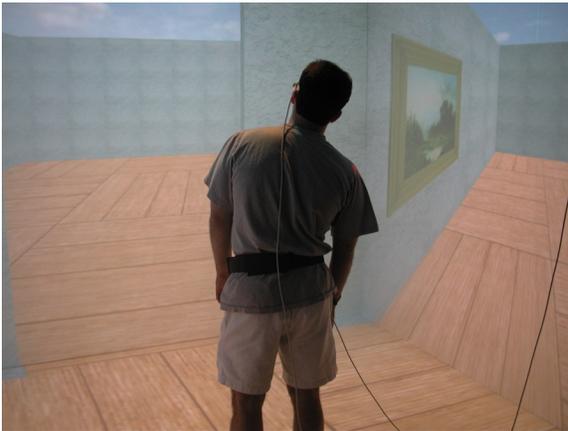
Scaled Up Step WIM

The user examines a scaled up version of the Step WIM.



The Step WIM

A user prepares to navigate with the Step WIM. The green sphere indicates his position in the miniature.



Leaning

A user leans to the right to examine a painting.



Interaction Slippers

An input device that allows the user to perform toe and heel tapping. Conductive cloth and a reconfigured wireless mouse are the two main components of the device.



Aladdin's Magic Carpet Ride



Virtual Pirates of the Caribbean



Aladdin HMD and seat



Pirate's Cannon and Motion Base



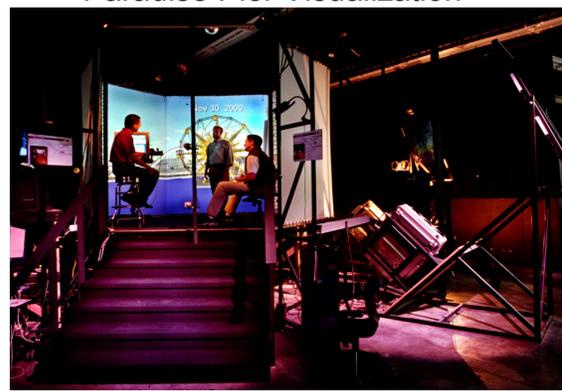
Hercules in the Underworld



Paradise Pier Visualization



Hercules Prototype CAVE



WDI R&D CAVE

Part II: Papers

HMD's, Caves & Chameleon: A Human-Centric Analysis of Interaction in Virtual Space

Bill Buxton & George W. Fitzmaurice
*Alias/Wavefront Inc.,
Toronto, Ontario
{buxton, gf}@aw.sgi.com*

Abstract

There are a various approaches to implementing Virtual Reality (VR) systems. The head mounted display (HMD) and Cave approaches are two of the best known. In this paper, we discuss such approaches from the perspective of the types of interaction that they afford. Our analysis looks at interaction from three perspectives: solo interaction, collaborative interaction in the same physical space, and remote collaboration. From this analysis emerges a basic taxonomy that is intended to help systems designers make choices that better match their implementation with the needs of their application and users.

Introduction

Immersive Virtual Reality (VR) was first suggested – as were so many other things – by Ivan Sutherland (1965). Practical working systems have now been with us for over a decade and have been written about extensively (e.g., Rheingold, 1991). If one includes the early work of Krueger (1983), they go back even further. The most well known approach to VR is that of the head mounted display (HMD) coupled with head tracking. With such systems, one typically is presented with a stereo binocular view of the virtual world, often with stereo audio. By virtue of tracking the viewing position (the head) and orientation in the physical world, the view and perspective of the virtual are consistent with what would experience in the physical world from the same actions.

In addition to tracking viewpoint, which is tied to what is displayed to the user, such systems also typically permit some means of input, such as a *dataglove* (Zimmerman, Lanier, Blanchard, Bryson & Harvill, 1987) or some other high degree of freedom input to support interaction with the displayed virtual world.

As the art progressed, alternative technical approaches to VR have emerged. Of these, we distinguish among three:

- *Head-Mounted VR*: systems as described briefly above, where one typically has a head-mounted wide-view stereo display coupled with head tracking, and some other means of input to support interaction.
- *Cave-based VR*: where some or all of the walls of a room are rear-projection stereo displays. The user wears glasses to enable viewing the stereo images, and there is a head-tracking mechanism to control what is projected (i.e., the view) depending on where the viewer is located and looking, as well as some mechanism for interacting with what is seen.
- *Chameleon-type VR*: which involves a hand held, or hand moved, display whose position and orientation are tracked in order to determine what appears on it. Furthermore, the display enables interacting with what appears on it.

Each of these types of VR system is discussed in more detail below. But the point of this paper is not to provide a history or enumeration of VR systems, *per se*.

VR, while expensive and still relatively new, is a powerful technology. It is being applied in a range of contexts ranging from entertainment to automotive design. But if one is going to engage the technology, then what path to follow, and why? What are the relevant dimensions? What are the pros and cons of each approach?

Providing some vocabulary and a framework to answering such questions is what motivates this brief discussion paper. After introducing each of the three classes of VR system, we discuss them in terms of their ability to support three types of interaction:

- *Solo*: where there is only one person interacting in the virtual space.
- *Same Place Collaboration*: where there is more than one user interacting in the virtual space, but they are physically situated in the same location.
- *Different Place Collaboration*: where there is more than one user interacting in the virtual space, but they are situated in different physical locations.

These are the key dimensions according to which we contrast the various approaches. It is obvious that other concerns such as cost, speed, fidelity, space requirements, etc. affect the choice of which technology to adopt. We will touch on some of these. But our overall objective is more modest: to shed some light on those dimensions that we feel we best understand.

Head-Mounted Display (HMD) VR

In HMD VR, the user mounts a stereo display, much like a pair of glasses that provide a view into the virtual world. The physical form of these "glasses" can range from something on the scale of a motorcycle helmet to a pair of sunglasses. Figure 1 illustrates one example of a HMD.

There is a great variety in display quality. The goal in the technology is to provide the widest field of view at the highest quality and with the least weight and at a reasonable cost. The reader is referred to Neale (1998) for a reasonably up-to-date survey of HMD technology.



Figure 1: Modern Inexpensive HMD: The General Reality CE-200W
(Photo: General Reality Corp.)

There is a range of high degree of freedom (HDOF) input devices that can be used in interaction with such systems. An overall directory of sources to input devices can be found in Buxton (1998). Furthermore, a number of classes of HDOF technologies are discussed in the contribution of Shumin Zhai (1998) in this special issue. Because of the typical mobility of the user (compared to desktop systems), however, most HMD systems use what Zhai calls a *flying mouse* class of device, often in conjunction with a data-glove type controller. In some cases, each hand is instrumented in order to support bimanual interaction.

The issue with virtually all HMDs is that the eyes are covered by the display. Consequently, one sees the virtual world at the expense of the physical one. Users cannot directly see their hands nor the devices that they are controlling. Similarly, they can not directly see objects or other people who are in their immediate physical environment. Therefore, in order to function, some representation of such entities from the physical world must appear in the virtual one. In order to use my hands, I most likely must see a representation of them. Likewise, in order to avoid bumping into a table, I must see a representation of it, and to avoid bumping into you, I must see an avatar, or some other representation of you.

In collaborative work a significant observation that emerges from this is that, visually, HMD VR treats those in the same and those in remote physical spaces the same (some would say equally poorly, since visually there is no advantage to "being there" physically).

There is an important caveat to raise at this juncture. Some researchers have found a way around the problem of seeing the physical world (such as objects, their hands, tools or other people) while wearing HMDs. One approach is to mount one or more video cameras onto the HMD and feed the signals to the displays (see Yoo and Olano, 1993; Azuma and Bishop, 1994; and State et al., 1996). The cameras function as surrogate eyes providing a view into the physical world onto which is superimposed a computer generated view of the virtual world. The result is much like a head's up display, and this approach to VR falls into the general category of *Augmented Reality (AR)*, since it enables the computer to augment our view of the physical world with additional information. See Feiner, MacIntyre and Seligmann (1993) for an example of Augmented Reality and its application.

One important application of this technology is in remote collaboration. As an example, take the case of a technician who needs guidance to repair a complex piece of equipment from an expert who is not physically there. Through the cameras mounted on the technician's HMD, the expert can remotely see what the technician is looking at. Conversely, using VR technology, the expert can point and indicate to the technician what to do. The guidance of the expert is superimposed on the technician's view of the equipment in the HMD, thereby enabling the repair to proceed.

Clearly the ability to support AR is an important attribute of HMD VR. However, since it is not in the mainstream of HMD VR, we will not discuss it further.

To compare the three VR approaches, we have defined a simple schematic to represent the relationship among the eyes, hands and display. Figure 2 shows a simple schematic of HMD VR systems. First, it shows that the eyes and display are both tightly coupled, physically and that their position is tracked. In addition, it shows that the hands are on the "far" side of the display. Finally, it shows that all three are physically coupled, and mobile within physical space.

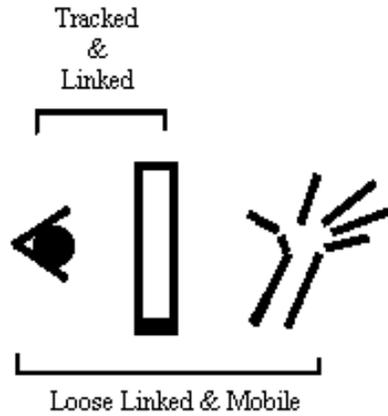


Figure 2: Schematic showing the relationship among the eyes, hands and display in HMD Style VR.

According to these criteria, and for the purposes of this paper, boom-mounted displays, such as illustrated in Figure 3, are a variation on HMDs, as opposed to a separate category (in contrast to the analysis of Cruz-Neira,, Sandin, DeFanti, Kenyon and Hart, 1992).



Figure 3: Fakespace BOOM3C boom mounted display (Photo: Fakespace, Inc.)

CAVES

A significantly different approach to VR, called *Cave VR*, was introduced by Cruz-Neira,, Sandin, DeFanti, Kenyon and Hart (1992). In this class of VR, the user functions within a room on which one or more of the surfaces (walls, floor, ceiling) is the display. An idealized representation of a cave is shown in Figure 4. This shows 4 sides of a 6-sided cave. In a cave, each of the displays is "tiled", in that together they provide a seamless omnidirectional view of the virtual scene. Furthermore, the displays are ideally stereo, and the operator views them through a set of lightweight transparent shutter glasses. The user's head position is tracked within the cave so that what is displayed preserves proper perspective, etc., in adapting to movements and change of location of gaze. That is, perceptually, the user sees the virtual scene in a manner consistent with it if it were real. And, as anyone who has seen a stereo movie knows, the objects in the virtual scene do not just appear on the cave walls and beyond. They can appear to enter into the physical space of the cave itself, where the user can interact with them directly.

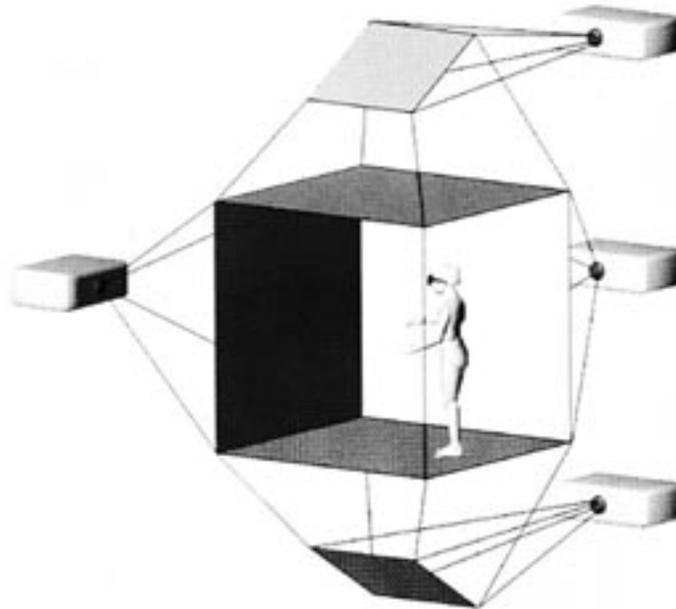


Figure 4: Schematic of an Idealized Cave VR System. Tiled rear projection stereo images appear on up to 6 faces of the room in which the operator works. In practice, most caves have 3-4 faces with projections. (Image from: Cruz-Neira,, Sandin, DeFanti, Kenyon and Hart, 1992).

As with HMD VR, manual interaction within the cave is typically accomplished with a HDOF device such as a "flying mouse" (sometimes coupled with speech recognition), in order to enable the operator to remain mobile within the space.

One area where caves differ from HMD VR is that, since the glasses are transparent, one can see the physical as well as the virtual world. Consequently, if you and I are both in the space, we can see each other as well as the virtual world. However, the way that we can share the scene has some distinct differences from HMD VR. Remember that what is displayed is determined by head tracking. If we are both in the cave, we both are viewing the same displays, preventing us from each having our own "point of view." (While we can both look at different things and different directions, we both do so as if from the perspective of the current location of the head tracker.) So the good news is, in the cave we really are presented with the same view. The bad news is, you have to see it from my location, or *vice versa*.

In remote collaboration, where two caves are linked, this constraint is softened since each cave can have a unique view, but everyone within a single cave must share the same one. But the advantage of being able to see each other in the context of the virtual scene is lost when collaborating across multiple caves. In remote collaboration one must resort to the same techniques used in HMD VR,—such as the use of avatars or some other representation, in order to see one's remote collaborators within the virtual space.

Finally, there is one potential problem that is unique to same-location collaboration in caves. In the everyday world, you and I may find ourselves on opposite sides of an object of interest or discussion. But what happens in a cave if the object of interest lies within the confines of the physical walls of the cave? If we are facing each other in a cave with a virtual object in between us, neither of us will be able to see the object as we are each blocking the screen on which it is being projected for the other person. Let us call this the "shadow effect."

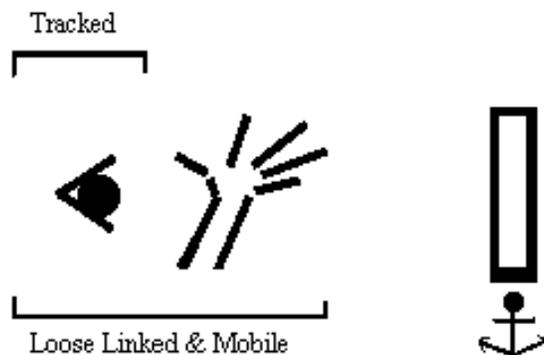


Figure 5: Schematic showing relationship among the eyes, hands and display in Cave Style VR.

As with HMD VR, in Figure 5 we characterize cave VR by means of a simple schematic. Here we illustrate that the eyes and hands are loosely coupled and mobile, and that the

display is anchored in a fixed position. Furthermore, it shows that the head is tracked and that the hands are visible, and are between the display and the eye.

Fitting into this characterization, are a number of other systems, which might therefore be considered "degenerate caves." One example would be large format projection displays such as the *ImmersaDesk* shown in Figure 6, developed at the Electronic Visualization Laboratory at the University of Illinois at Chicago (Czernuszenko, Pape, Sandin, DeFanti, Dawe and Brown, 1997). This is essentially a small 1-sided cave.



Figure 6: The ImmersaDesk VR System. A Large format rear-projection flat stereo display (Photo: Electronic Visualization Laboratory at the University of Illinois at Chicago)

Another example would be what Ware and Booth (1993) called Fish tank VR. These are typically CRT systems which incorporate head-tracking, and present a perspective view (often not stereo), based on the user's head position. Such systems can be thought of as very small format one-sided caves (tunnels?) with a consequently limited field of view and range of mobility of the user.

Actually, small format caves have been built, showing that you don't have to be able to walk around in a cave for the technology to be of value. The *Cubby* system developed at the Technical University of Delft in The Netherlands is one such example (Djajadiningrat, 1998; Djajadiningrat, Smets & Overbeeke, 1997).

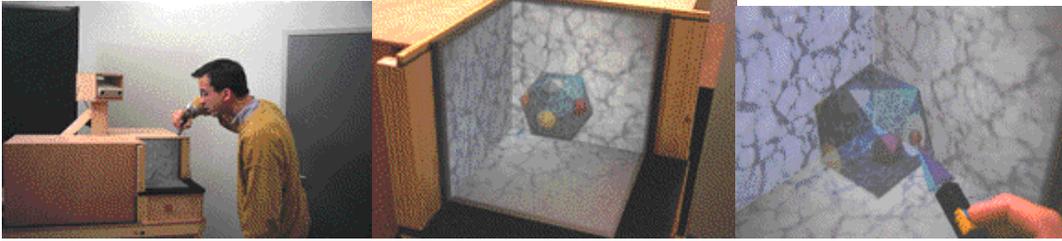


Figure 7: The Cubby System: A Small 3-Sided Cave
(Djajadiningrat, 1998; Djajadiningrat, Smets & Overbeeke, 1997).

Finally, flight and driving simulators, which involve a vehicle in a space (often only partially) surrounded by rear projection screens, would also fall into this category. The display is often not stereo, and it may not be flat. And the user is typically not mobile, being confined to the vehicle. However, the basic relationship among the view, hands and display are consistent with the cave approach.

CHAMELEON-Style VR

The third and least well known approach to VR that we will discuss was introduced by Fitzmaurice (1993) in his *Chameleon* system. This can be thought of as hand-held VR. In the Chameleon system, the image appeared on a small display held in the palm of the hand. In this case, what appeared on the screen was determined by tracking the position of the display, rather than the head of the user.

One way to think about the Chameleon approach is as a magnifying glass that looks onto a virtual scene, rather than the physical world. And while the display is small, and certainly does not give the wide angle view found with the cave approach, the scene is easily browsed by moving the lightweight display, as shown in the right hand image of Figure 8.

This movement of the display actually takes advantage of a subtle but powerful effect in human visual perception. With respect to visual perception, Newton was wrong about the equivalence of relative motion. That is, moving a scene on a fixed display is not the same as moving a display over a stationary scene. The reason is rooted in the persistence of images on the retina, formally known as the "Parks Effect," (Parks, 1965). Much like moving the cursor often leaves a visible trail on a screen, moving the Chameleon display across the field of vision, and updating the view with the motion, can leave an image of the larger scene on the retina. Hence, if the display can move, the effective size of the virtual display need not be the same as the physical size. (If you remain confused, think about the effect of drawing a pattern on a wall by quickly moving a laser pointer. Here, a whole pattern is displayed even though only one point is illuminated at any given time. The image is in your eye, not on the wall. Such is the human visual perceptual system, and Chameleon-like VR can take advantage of it.)

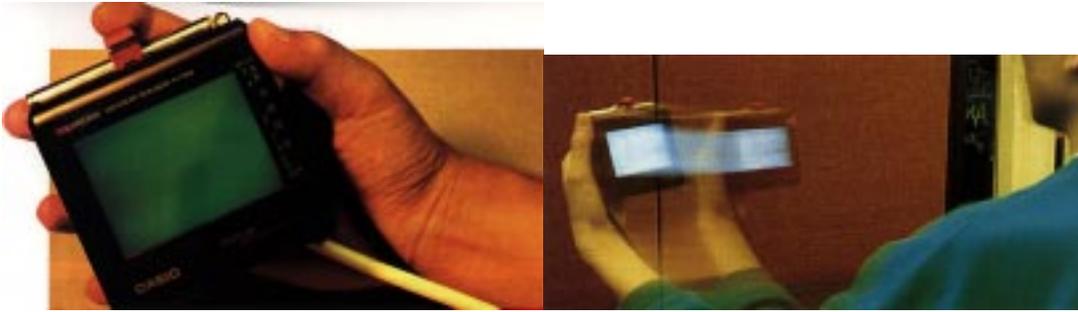


Figure 8: Chameleon Palm-held VR System. A monocular image is presented on a palm sized portable display. The display has position and orientation tracking so what is displayed is determined by the display position. (It is like a virtual magnifying glass). The display also incorporates some manual controls. (Photo from Fitzmaurice, 1993)

Interaction with this class of display tends to be based on devices such as buttons or (as seen in the next example) a touch screen coupled directly with the display. That is to say, the display device serves for both input and output. In no cases, to our knowledge, has stereo display been used with this class of system, although one can imagine achieving this using the same kind of shuttered glasses employed in cave systems.

Like cave systems, in Chameleon-like VR, one has an unobstructed view of people and objects in the physical world. However, unlike the cave but as with HMD VR, in collaborating with others in the same physical space, each user has their own view. And yet, it is easy to have a mechanism for sharing a view without disorienting the other viewers, since orientation is mainly determined by one's orientation in physical space. (Contrast this with switching views in either cave or HMD VR.)

On the other hand, Chameleon VR shares the same problem as both HMD and cave VR in establishing a sense of presence of others in collaboration involving different physical locations.

As with the other techniques, we can characterize Chameleon-like VR schematically. Figure 9 illustrates the tight coupling of the hand(s) with the display, as well as the tracking of the display, and the mobility (modulo any tethering) of all three.

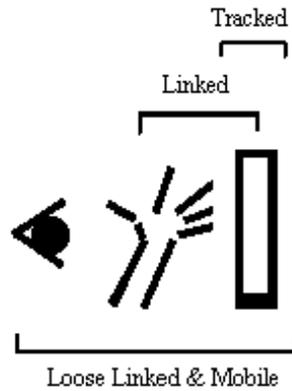


Figure 9: Schematic showing relationship among the eyes, hands and display in Chameleon Style VR.

There have been other examples that have taken the Chameleon-like approach. For our purposes, one of the most interesting was developed by Art+Com (1998) to enable the public to view a virtual version of the new Daimler-Benz A-class vehicle at the IAA motor show in Frankfurt, September of 1997. This is illustrated in Figure 10.



Figure 10: Art+Com Virtual Car Display. This system is essentially a larger format display version of Chameleon. A Counter-balanced boom constrains the display movement as well as supports its weight. (Photo: Art+Com)

In this example, the display was larger than in the original Chameleon system. Rather than hand-held, it was supported by a counter-balanced boom. While mechanically not unlike the Fakespace boom seen previously in Figure 3, conceptually this system is quite distinct. It very much falls into the Chameleon-class of VR by virtue of the relationship of the hands to the display, and the user's simultaneous visibility and awareness of the surrounding physical space.

In this example, the system was on a scale to enable the car to be viewed on a 1:1 scale. The user could walk through and view the virtual car with the help of a flat screen (LCD) attached to a swivel arm. What this example demonstrates is how the technology for interacting with the virtual space can be integrated seamlessly into the display. This is shown in Figure 11, which illustrates how a touch screen on the display was used to select things such as the colour of the vehicle or fabric of the upholstery.



Figure 11: Art+Com VR Control: Note that the display in the previous photo is a touch screen that enables the operator to interact with the image. (Photo: Art+Com)

Finally, like HMDs, Chameleon-like systems have the ability to support augmented reality. In his paper, for example, Fitzmaurice (1993) showed how location tracking not only told the device where it was physically, but also relative to other devices, or people. Brought close to a map, for example, it could give additional information about the region that it was close to. Or, brought beside a complex piece of machinery, by being aware of the fact, it could give valuable information about how to use or repair the device.

Some researchers, Rekimoto and Nagao (1995), for example, have augmented Chameleon-like devices further and added video cameras in a manner similar to those discussed in the section on HMDs. Using this approach, the computer generated information can, likewise, be superimposed over a view of the physical world, with the same benefits discussed with HMDs.

SUMMARY & CONCLUSIONS

In the preceding we have surveyed three distinct approaches to VR. We have attempted to describe each in terms of properties that might influence their suitability for different types of applications. In particular, we have emphasized properties that emerge in different forms of collaboration. These are summarized in Table 1.

		Same-Place Collaboration	Different-Place Collaboration	Support AR?
	Solo			
HMD	<ul style="list-style-type: none"> • see virtual space only • hands and tools by virtual representation only (but see support for AR column) 	<ul style="list-style-type: none"> • see from personal viewpoint • awkward shared viewpoint • only see others as avatar, for example (but see support for AR column) 	<ul style="list-style-type: none"> • see from personal viewpoint • awkward shared viewpoint • same place and different place collaborators treated the same, as avatars 	<ul style="list-style-type: none"> • yes, if HMD coupled with video camera(s) for example. Then local objects, hands & people visible.

<p>Cave</p>	<ul style="list-style-type: none"> • see virtual and physical space • hands and tools visible 	<ul style="list-style-type: none"> • see from viewpoint of another (but possibly different view direction) • see others in physical space • potential shadow effect blocking view of object of interest 	<ul style="list-style-type: none"> • only one viewpoint per site • only see remote participants as avatars, for example 	<ul style="list-style-type: none"> • no
<p>Chameleon</p>	<ul style="list-style-type: none"> • see virtual and physical space • hands and tools visible 	<ul style="list-style-type: none"> • see from personal viewpoint • potential non-disruptive shared viewing • see others in physical space 	<ul style="list-style-type: none"> • see from personal viewpoint only see remote participants as avatars, for example 	<ul style="list-style-type: none"> • yes, with or without video camera to augment display

Table 1: Properties of VR Systems for Various Numbers and Distribution of Users

Obviously, other factors will also affect what technology is adopted. Cost is always an issue. So is the question of the amount of space, and any specialized environments required. And even within type, there is a broad range of variation, in image quality, responsiveness, etc.

But in many cases, it may be that more global human factors are most important. By way of example, consider an automotive design studio that wants to use VR technology for design reviews. Cave technology can and has been used to good effect. However, the quality has to be balanced with the fact that there typically isn't a cave in every studio. Rather, the cave is most commonly a shared resource in a different part of the building. It has to be booked and data transferred and set up. While this structure can support formal reviews, it does not lend itself to casual or spontaneous reviews by management, customers or designers. That is to say, social issues might be the determining factor in choosing something like a Chameleon VR system, *even if the fidelity does not match that of the alternative approaches.*

VR technologies are expensive and not well understood. In our opinion, there is no "right approach" without a careful analysis of user, task and context (physical and social). Hopefully, the concepts outlined in this paper make some progress in paving the path to an understanding of the issues that will support such decisions. In the meantime, the authors welcome comments, suggestions and questions.

ACKNOWLEDGEMENTS

The research underlying this paper has been supported by Alias|Wavefront, Inc. and Silicon Graphics, Inc. This support is gratefully acknowledged. Also thanks for Thomas Baudel and Michael Mills for valuable suggestions and help.

REFERENCES

ART+COM (1998). <http://www.artcom.de/projects/vrf/welcome.en>

Azuma, R. and Bishop. G. (1994). Improving static and dynamic registration in an optical see-through HMD. *Proceedings of SIGGRAPH'94*, 197-204.

Buxton, W. (1998). *A Directory of Sources to Input Technologies*.
<http://www.dgp.utoronto.ca/people/BillBuxton/InputSources.html>

Cruz-Neira, C., Sandin, D.J., DeFanti, T.A., Kenyon, R.V., and Hart, J.C. (1992). The CAVE: Audio Visual Experience Automatic Virtual Environment, *Communications of the ACM*, 35(6), 65-72.

Czernuszenko, M., Pape, D., Sandin, D., DeFanti, T., Dawe, G. L., and Brown, M. D. (1997). The ImmersaDesk and Infinity Wall Projection-Based Virtual Reality Displays. *Computer Graphics*, 31(2), 46-49.

Djajadiningrat, J.P. (1998). *Cubby: What You See is Where You Act*, PhD Thesis, Technical University of Delft, The Netherlands.
<http://www.io.tudelft.nl/research/IDEATE/cubby/cubby.html>

Djajadiningrat, J.P., Smets, G.J.F. & Overbeeke, C.J. (1997). Cubby: a multiscreen movement parallax display for direct manual manipulation, *Displays* 17, 191-197.

Fakespace, Inc., 241 Polaris Ave. Mountain View, CA 94043 USA.
<http://www.fakespace.com/>

Feiner, S., MacIntyre, B. & Seligmann, D. (1993). Knowledge-Based Augmented Reality. *Communications of the ACM*, 36(7), 53-62.

Fitzmaurice, G.W. (1993). Situated Information Spaces and Spatially Aware Palmtop Computers. *Communications of the ACM*, 36(7), 38-49.

Krueger, Myron, W. (1983). *Artificial Reality*. Reading: Addison-Wesley.

Neale, D. (1998). Head-Mounted Displays: Product Reviews and Related Design Considerations, *Hypermedia Technical Report HCIL-98-02*, Human-Computer Interaction Laboratory, Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, VA 24061-0118. <http://hci.ise.vt.edu/~hcil/htr/HCIL-98-02/HCIL-98-02.html>

Parks, T.E. (1965). Post Retinal Visual Storage, *American Journal of Psychology*, 78, 145-147.

Rekimoto, J. and Nagao, K. (1995). The world through the computer: computer augmented interaction with real world environments. *Proceedings of UIST'95*, 29-36.

Rheingold, H. (1991). *Virtual Reality*. N.Y.: Summit.

State, A., Hirota, G., Chen, D.T., Garrett, W.F. and Livingston, M.A. (1996). Superior augmented reality registration by integrating landmark tracking and magnetic tracking. *Proceedings of SIGGRAPH'96*, 429-438.

Sutherland, I. (1965). The Ultimate Display. *Proceedings of IFIP 65, Vol. 2*, 506-508, 582-583.

Ware, C., Arthur, K. & Booth, K. (1993). Fish tank virtual reality, *Proceedings of InterCHI '93*, 37-42.

Yoo, T.S., Olano, T.M. (1993). Instant Hole (Windows onto Reality). University of North Carolina at Chapel Hill Technical Report TR-93-027.
<http://www.cs.unc.edu/Research/graphics/pubs.html>

Zhai, S. (1998). User Performance in Relation to 3D Input Devices. To appear, *Computer Graphics Quarterly*, November 1998.

Zimmerman, T.G., Lanier, J., Blanchard, C., Bryson, S. & Harvill, Y. (1987). A Hand Gesture Interface Device, *Proceedings of CHI+GI '87*, 189-192.

FLEX AND PINCH: A CASE STUDY OF WHOLE HAND INPUT DESIGN FOR VIRTUAL ENVIRONMENT INTERACTION

JOSEPH J. LAVIOLA JR. and ROBERT C. ZELEZNIK

Brown University Site of the NSF Science and Technology Center
for Computer Graphics and Scientific Visualization
PO Box 1910, Providence, RI 02912 USA

ABSTRACT

We present a discussion of design issues involving whole hand input in virtual environments. In many cases, whole hand input devices limit the types of interaction that the user can perform in the virtual world due to the nature of the device. One possible approach to alleviate these limitations is to provide hybrid input devices which enable the user to combine information generated from two different whole hand input devices. In this paper, we describe our Pinch Glove like input device which is used as a tool to augment bend-sensing gloves for object manipulation and menu selection as well as a method to test and evaluate different hand postures and gestures that could not be developed with a single whole hand device.

KEYWORDS: Human-Computer Interaction, Virtual Environments, 3D Graphics Applications, Conductive Cloth, Flex and Pinch Input

INTRODUCTION

There have been a number of different approaches for interacting in virtual environments. In general, these approaches have attempted to solve small interface problems in isolation without incorporating them into complete interface solutions. For example, consider the Head Crusher object selection technique[1] which allows the user to very naturally select and manipulate 3D objects with just one hand by positioning the thumb and forefinger around a 2D image of the desired object. To actually use this technique for object selection, the user must hold and press a button in their other hand.

Another important reason why many of these interaction techniques solve small problems in isolation has to do with the nature of the available input devices used. In most cases, individually specialized input devices work well for the interaction techniques they were designed for. However, they have difficulty mapping combinations of techniques or

applying different techniques due to their inflexibility. For example, consider bend-sensing gloves which report continuous joint angle measurements of the fingers. With these devices, relatively slow and complicated posture and gesture recognition techniques must be used to generate discrete events that would otherwise be trivial with a button press.

In order to increase the flexibility of input devices, to extend existing virtual environment interaction techniques, and to create more robust virtual environment interfaces, we believe that hybrid interfaces – interfaces that seamlessly combine input devices and interaction techniques – will provide a more flexible and robust method of interacting in virtual environments. With Flex and Pinch input, we have developed a hybrid input device which combines continuous joint angle measurements and discrete pinch button input. By having this combination, we can improve on a number of existing virtual environment interface techniques and develop new ones.

ORGANIZATION

The remainder of this paper is organized in the following manner. The next section describes previous work related to Flex and Pinch followed by a description of the components and design issues in developing our interface hardware. Then we describe a number of interface techniques that use Flex and Pinch input. Finally, the last two sections provide areas for future work and a conclusion.

PREVIOUS WORK

There are two basic approaches to using whole hand input in virtual environments. First, the non-invasive approach uses vision-based tracking[2] so the user is not physically attached to the computer. Second, the invasive approach uses a glove-based device or devices to extract information from the hands. In each approach, we can extract two different types of data, namely geometrical data and topological data. Geometrical data represents information about the hand's shape while topological data provides information

about how the fingers touch each other and other parts of the hand. Although a non-invasive approach maybe preferred, it is difficult to extract both geometrical and topological information due to problems with computer vision such as occlusion. Therefore, we focus on the invasive approach.

With the invasive approach, two types of glove-based input devices have been developed. The first, bend-sensing gloves[3][4][5], measure finger joint movement, and second, the Pinch Glove[6][7], detect electrical contacts between each of the finger tips. Unfortunately, bend-sensing and pinch gloves have faults when used in isolation. Bend-sensing gloves are good at extracting geometrical information which enables them to represent the user's hands in the virtual environment. They can be used to mimic interface widgets such as sliders and dials[8], but do not have useful methods for signaling the activation or deactivation of the widget. Bend-sensing gloves are also used in conjunction with hand posture and gesture recognition, but it can be difficult to determine when one gesture begins and another ends without applying constraints to the users gesture space[9]. Conversely, Pinch gloves provide a series of button widgets that are placed on each finger tip which allows for the extraction of topological data for interactions such as pinching postures. However, they have no way of determining the flexing of the fingers and they make it difficult to represent the hand in a virtual environment.

There have been few attempts to combine the two types of information that each type of data glove provides. With the exception of Grimes' Digital Data Entry Glove which was developed specifically for entering text using the Single Hand Manual Alphabet[10], there has been little work done with combining discrete and continuous whole hand input devices to extract both geometrical and topological data simultaneously.

OVERVIEW OF APPROACH

In order to develop an interface that combines both geometrical and topological data, we built a hardware prototyping system for testing and evaluating different interface designs¹. The hardware system provides a number of benefits in that it employs a plug and play strategy for quickly adding and removing button widgets or their components. Our system enables users to incorporate up to 16 cloth sensors in a wearable interface. Conductive cloth[11] sensors provide two important functions: first, each sensor knows when it comes in contact with another sensor and specifically which other sensor it contacts, second, the nature of the cloth lends itself for use on gloves or clothing.

Using our prototyping system, we have constructed a device based on the Fakespace Pinch Glove[6]. As a hardware input device, it provides more functionality than the

¹Appendix A provided a description of the electronics and the various components used for building our hardware system.

Pinch Glove since it uses eight cloth buttons instead of five which allows for more button combinations. In general, five of these cloth buttons can be placed around each of the finger tips, while the other three can be placed arbitrarily about the hand². Using this device, we augment existing bend-sensing gloves to create Flex and Pinch input (see Figure 1).



Figure 1: The Flex and Pinch input system. Although a CyberGlove[4] is shown, any bend-sensing glove can be used.

INTERACTION TECHNIQUES USING FLEX AND PINCH INPUT

With Flex and Pinch input, we can improve on a number of existing techniques for selecting objects in virtual environments and create new techniques that could not be developed without the combination of geometrical and topological data. For example, one of the major problems with the image plane interaction techniques such as the head crusher, sticky finger, lifting palm, and framing hands object selection techniques[1] is that the user cannot activate the selection with the primary hand. As a result, the user requires an additional, separate input device for triggering the selection operation.

Flex and Pinch input provides a simple yet effective and seamless method for starting and stopping object selection by placing the cloth buttons in appropriate places on the users primary hand. For example, with the head crusher technique, we can place the cloth buttons on the thumb and middle finger so when the user positions the thumb and forefinger around the object a middle finger to thumb contact will signal the object should be selected. Another button press would signal the release of the object. The cloth contacts can be placed in other positions such as on the middle finger and on the palm by the base of the thumb or on the right side of the index finger and the left side of the middle finger. In a similar manner, cloth contacts are placed on

²This presents one of many possible combinations for placement of the cloth buttons. The device could have be worn with anywhere from two to 16 cloth buttons of any shape or size. This presents a clear advantage over other inflexible input devices.

the hand for the sticky finger and lifting palm techniques to start and stop object selection while cloth contacts are placed on both hands for the framing hands selection technique. Figure 2 shows the Head Crusher technique with placement of the cloth contacts between the forefinger and middle finger.

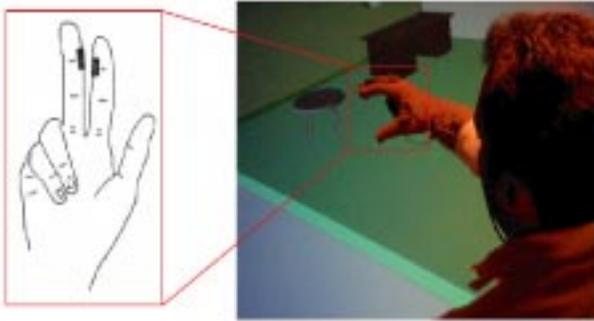


Figure 2: A user wearing the Flex and Pinch input device is about to invoke the Head Crusher object selection technique on a round table. By placing his middle and index finger together, the user can activate the selection operation and move the table.

Another method that has been used for selecting objects in virtual environments is to cast a laser into the scene from the users hand to select a given object[12]. As with the image plane techniques, the problem with laser pointing is it is difficult to start and stop the selection with only one input device. For example, one laser pointing object selection method uses a point and clutch posturing mechanism to select objects in a virtual environment where clutching is performed by flexing the thumb[13]. The problem with using this clutching mechanism is that in order to achieve robust recognition, the user must make postures using extreme configurations of the hand which puts undo strain on the two tendons in the thumb. Using Flex and Pinch input we can alleviate this problem by placing cloth contacts on the thumb and on the right side of the middle finger as shown in Figure 3. This provides a much more natural movement and puts no strain on the thumb tendons³.

Bend-sensing gloves have the capability of being used as analog sliders since these gloves report continuous measurements of the joint angles in the hand. However, used in isolation, it can be difficult to determine when the user wants to actually use one of the fingers as a slider widget. Using Flex and Pinch input, a seamless transition between the discrete events from the cloth contacts and the continuous updating from the bend sensors can be made which provides a mechanism for activating and deactivating the sliders when needed. For example, we can cycle through

³One could argue that the user could make a posture that is identical to the user's hand configuration when using Flex and Pinch input. However, hand gesture and posture recognition is not perfect, and if the hardware is working properly, the pinching mechanism will provide 100 percent accuracy.

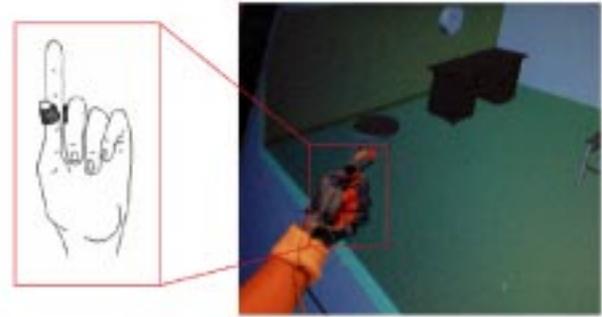


Figure 3: A user pointing at and selecting a desk in the virtual environment. The user makes the selection by pressing the thumb to the right side of the middle finger.

menu items with a finger⁴. A button press creates the menu and as the button is held, the user can cycle through the menu items by flexing or extending a finger. If the user does not wish to select an item, they need to release the button when their finger is fully extended or fully flexed. We are currently exploring how many menu items a user can easily invoke using this technique. Using the same configuration we also can change an object's scaling, translational, and rotational parameters.

Finally, an important benefit of using the pinch component of Flex and Pinch is that it gives application developers a method to test out different hand postures and gestures. In many cases, when a developer wants to test a new hand posture or gesture, they have to retrain their gesture recognition algorithms[14] which is time consuming. The pinch component of Flex and Pinch input allows the developer to quickly move cloth contacts from one part of the hand to another without having to change any software components or restart the application. This allows the application developer to quickly test the feeling and ergonomics of certain hand postures and gestures. Also, with the ability to move the cloth contacts anywhere on the hand, we can create whole hand interfaces that could not be implemented with a bend-sensing glove or the Pinch Glove used in isolation.

FUTURE WORK

There are a number of areas of future work that must be researched to determine if these hybrid interfaces provide virtual environment interaction methods that are useful. We plan to continue developing new hybrid input devices and exploring how they can give users better performance in virtual environment applications. In order to do this, extensive user studies are required to evaluate whether our

⁴In this case, one cloth contact is placed on the thumb while the second is placed on the left side of the forefinger between the Proximal Interphalangeal and Metacarpophalangeal joints.

interaction techniques are indeed better than existing techniques. We believe that the input devices and interaction techniques we have developed are just the tip of the iceberg. As a result, it is important to continue to research how they can be applied to different interactions in and out of virtual environments.

Another area of work that needs further exploration is whether or not users prefer a wearable interface solution over a less obtrusive solution such as computer vision-based interaction. Although the invasive approach provides more functionality since occlusion problems can occur with vision-based tracking, typically users do not want to be physically connected to the computer. One compromise between unobtrusive interfaces and increased functionality is using wireless input devices.

CONCLUSION

In this paper we have presented a case study on whole hand input design issues for virtual environment interaction using Flex and Pinch input. Using our custom built hardware prototyping system, we have developed a multi-purpose button based input device that can be used to develop seamless, hybrid interfaces by augmenting devices that produce continuous input events. With Flex and Pinch input, we can improve on existing virtual environment interaction techniques such as the image plane object selection techniques[1]. We also can develop novel hand postures and gestures that could not otherwise be developed with a device that generates purely geometrical or topological data. With further study and research, it is our goal to make the geometrical/topological approach a powerful metaphor for interaction in virtual environments.

ACKNOWLEDGMENTS

Special thanks to Timothy Rowley for helpful discussions during the hardware design and implementation, Brian Perkins for providing the electronics implementation, and Christine Waggoner for invaluable assistance with creating Flex and Pinch. This work is supported in part by the NSF Graphics and Visualization Center, International Business Machines, Advanced Networks and Services, Alias/Wavefront, Autodesk, Microsoft, Sun Microsystems, and TACO.

APPENDIX A

This appendix provides information on the design and implementation of our custom built hardware for quickly prototyping and testing hybrid, whole hand input devices.

DESIGN AND IMPLEMENTATION OF ELECTRONICS

The Microchip PIC processor[15] was chosen as the primary means of interfacing the touch sensors with the rest of the system. The low cost and simple programming of these chips made them suitable for the task. The 16C63[16] provided a UART for serial communications with the workstation, and enough I/O pins to allow the touch sensors to be monitored without extra glue logic. The output pins of the micro-controller were protected from electrostatic discharge with a resistor capacitor network. Additionally, an rs232 driver chip was needed to step the five volt output of the PIC to rs232 line levels.

All 163 possible non-redundant contact possibilities between pairs of wires are reported by separate keycodes. It is up to the microcode driver to report separate keycodes for wire connections while the driver on the workstation infers contacts between more than two wires. For example, if contacts one, two, and three are touching, the microcontroller will report that one and two are touching by issuing one keycode, one and three are touching by issuing another keycode, and that two and three are also touching by issuing a third keycode. It is up to the driver software to determine that there are actually three wires that are all touching. This lowers the amount of memory needed on the microcontroller, and makes the software simpler and faster.

PARTS LIST

PART	USAGE
PIC16C63	8 bit microcontroller with built in UART primary interface chip
16x 20K ohm resistors	pull up resistors
16x 2K ohm resistors	protection resistors
16x 1000pF capacitors	protection capacitors
LT1081	RS232 driver/receiver converts 5 volt PIC output to RS232 levels

ELECTRONICS PSEUDOCODE

This pseudocode represents the code for the PIC on the electronics box. Each short possibility has a byte allocated to it to represent the status (short or unshort) and a timer to determine whether the short has lasted long enough to transmit. This implementation cuts down on noise and bouncing problems.

Algorithm 1

```
1. initializeMemory()
2. for each pin
3.     do set a voltage on pin;
4.     for each (otherpin > pin)
5.         do check for voltage on otherpin;
6.         if (pin status changed)
7.             increment keycode timer;
8.         if (timer expired)
9.             set keycode status;
10.            transmit status change;
```

REFERENCES

- [1] J.S. Pierce, A.S. Forsberg, M.J. Conway, S. Hong, R.C. Zeleznik, and M.R. Mine. Image Plane Interaction Techniques in 3D Immersive Environments. *Proceedings of the 1997 Symposium on Interactive 3D Graphics*, 1997, 39-44.
- [2] Y. Kuno, T. Ishiyama, K. Jo, N. Shimada and Y. Shirai. Vision-Based Human Interface System: Selectively Recognizing Intentional Hand Gestures. In *Proceedings of the IASTED International Conference on Computer Graphics and Imaging*, 1998, 219-223.
- [3] Nissho Electronics Corporation. Introduction to SuperGlove. Tokyo, Japan, 1997.
- [4] Virtual Technologies. CyberGlove™ User's Manual. Palo Alto, California, 1993.
- [5] T.G. Zimmerman, J. Lanier, C. Blanchard, S. Bryson, and Y. Harvill. A Hand Gesture Interface Device. In *Proceedings of CHI+GI'87 Human Factors in Computing Systems and Graphics Interface*, 1987 189-192.
- [6] Fakespace Pinch™ Glove System Installation Guide and User Handbook, Mountain View, California, 1997.
- [7] D.J. Mapes and M.J. Moshell. A Two-Handed Interface for Object Manipulation in Virtual Environments. *PRESENSE Teleoperators and Virtual Environments*, 1995, 4(4):403-416.
- [8] D.J. Sturman and D. Zeltzer. A Survey of Glove-based Input. *IEEE Computer Graphics and Applications*, 1994, 14(1):30-39.
- [9] D.J. Sturman. *Whole Hand Input*. Ph.D. dissertation, Massachusetts Institute of Technology, 1992.
- [10] G. Grimes. Digital Data Entry Glove Interface Device. Bell Telephone Laboratories, Murray Hill, New Jersey. US Patent Number 4,414,537.
- [11] S. Mann. Smart Clothing: The Wearable Computer and WearCam. *Personal Technologies*, Volume 1, Issue 1, March, 1997.
- [12] K. Hinkley, R. Pausch, J.C. Goble, and N.F. Kassel. A Survey of Design Issues in Spatial Input. *Proceedings of the ACM Symposium on User Interface Software and Technology*, 1994, 213-222.
- [13] J.J. LaViola Jr. A Multimodal Interface Framework For Using Hand Gestures and Speech in Virtual Environment Applications. To appear in *Lecture Notes In Artificial Intelligence: The Gesture Workshop'99*, Springer-Verlag, 1999.
- [14] S. Fels and G. Hinton. Glove-TalkII: An Adaptive Gesture-to-Format Interface. *Proceedings of CHI'95 Human Factors in Computing Systems*, 1995, 456-463.
- [15] <http://www.microchip.com/10/Lit/PICmicro/index.htm>.
- [16] <http://www.microchip.com/10/Lit/PICmicro/16C6X/index.htm>.

INVESTIGATING COORDINATION IN MULTIDEGREE OF FREEDOM CONTROL I: TIME-ON-TARGET ANALYSIS OF 6 DOF TRACKING

Shumin Zhai
IBM Almaden Research Center
San Jose, California, USA

John W. Senders
University of Toronto
Toronto, Ontario, Canada

In these two companion papers, methods developed in a series of studies in the 1940's and 1950's are applied to the analysis of 6 DOF control devices used in modern human machine systems such as teleoperation and virtual environments. Contrary to the early studies, the current work showed that the simultaneous time-on-target in multidegree of freedom tracking was higher than the product of component time on target scores. The distribution of linear correlation coefficients between the tracking errors of different degrees of freedom tended to be skewed towards the positive values. These results suggested that subjects' discoordination in early multidegree of freedom tracking studies was likely due to the limitation of human machine interfaces at that time. With well designed interfaces, subjects exhibited more coordinated trials than discoordinated trials in multidegree of freedom tracking.

INTRODUCTION

It has always been interesting to read about early efforts in the exploration of a new science. The late 1940's and early 1950's was a time of intense interest in human control of systems. This interest was driven in part by developments in Control Theory and Information Theory and in part by the demands of the sponsoring agencies, chiefly military. Behavioral theorists saw in the emerging engineering approaches new ways of formalizing human behavior. The sponsors' concern was largely one of whether a target would be hit. Ease of computation and high face validity led to the use of "time-on-target" as a measure of "tracking" performance. It was natural that the study of multiple degree-of-freedom (DOF) tasks would arise (e.g. Ellson, 1947; Senders, Christensen, & Sabeh, 1955; Senders, Wallis, & Senders, 1956). Poor performance in multiple DOF control was attributed to lack of "coordination". It was not clear whether "coordination" was a purely human characteristic or arose from interaction with equipment design.

The growing number of applications of teleoperation, virtual environments and other 3D human machine systems has renewed research interest in coordination. Manual controllers have been designed to allow translational and rotational manipulation in 3D space with 6 DOF (see Brooks & Bejczy, 1985; Jacobus, Riggs, Jacobus, & Weinstein, 1992, and Zhai, 1995 for reviews). How well human operators can handle all 6 DOF has not been satisfactorily resolved especially with respect to control with one hand. Rice, Yorchak and Hartley (1986) observed that controlling 6 DOF with one hand is difficult. Some teleoperation systems, such as the Shuttle Remote Manipulator, also known as the "Canadarm", require two-handed operation: one hand for rotation control and the other for translation control. O'Hara (1987) contradicted Rice's observation, however, and found no differences between two 3 DOF

controllers and one 6 DOF controller. To base the design and selection of multidegree of freedom control interfaces on a firm ground, human coordination in using 6 DOF control devices is a fundamental problem that must be addressed.

A quantitative (and preferably analytic) measure of coordination that also satisfies intuitive understanding of the concept is of critical importance. Ellson (1947) derived independence measures of percent time-on-target scores (TOT) on the DOFs (azimuth, elevation, and range) of the Pedestal Sight Manipulation Test (PSMT). He recorded *simultaneous* TOT's (STOT) in all pairs of dimensions as well as all three at once, in addition to TOT's in each of the component dimensions. He then compared STOT scores with the products of the component TOT scores. His argument was that if the percent STOT was *equal* to the product of the component TOTs, then the components may be considered independent (uncorrelated). If greater, they were positively correlated; if less, negatively correlated. Ellson found that the tracking of most subjects was characterized by a slightly negative relationship: STOT scores were slightly less than the products of the component TOT scores. In other words, there was some tendency for the subjects to be off target in one dimension when on target in another dimension. Gardner found that when subjects used a joystick control with a cross-pointer display (Gardner, 1950) the function $STOT - (TOT_x) \cdot (TOT_y)$ was not significantly different from zero.

Senders (Senders et al., 1956) extended Ellson's approach. He placed the scores on a two DOF task into a 2 x 2 matrix of STOT and component TOT's and computed the phi-coefficient, ϕ . ϕ is an approximation of the product-moment correlation coefficient of two arbitrarily dichotomized continuous variables. Senders and colleagues found that subjects' tracking in a two DOF task which required manipulation of two knobs to control a pointer on a dial, produced more negatively correlated than positively correlated trials and, with continued practice, actually produced larger negative correlations.

If one operator could not coordinate tracking in two axes, it might be better to assign the tracking task to two operators. Such an issue was studied in (Senders et al., 1955) which had three groups of subjects participating in a two dimensional tracking task. Members in Group I tracked both dimensions. The second group had several *teams of two* subjects performing the tracking task; each subject operated one dimension of the task and information about both dimensions was displayed to the team. The third group worked the same way as in Group II but each subject of the team was presented with only the dimension that he was tracking. Results showed that Group I performed at a much lower level of performance than the teamed groups (as measured by component TOT's as well as STOT's).

These early studies seemed to suggest a rather pessimistic view of one operator's ability to coordinate two or more dimensions. Note, however, that the negative correlations between dimensions were not necessarily due to subjects' inability to coordinate control actions, but possibly an artifact of the control and display interfaces available at the time. In Ellson's study, the PSMT did not permit "the correction of simultaneous errors in several dimensions by a single well-coordinated movement." (Ellson, 1947). In Senders' study, the displays were separated and each control was operated by one hand. In Gardner's study (Gardner, 1950), although a joystick and a cross pointer display were used, "opportunity did not exist for a single corrective movement for errors in both dimensions". Whether operators could coordinate multiple DOF control, if appropriate interfaces were provided, remains an open question. We have turned a PC into a virtual time machine and gone back to do what could not reasonably have been done then.

Modern interfaces in teleoperation and 3D display systems, although involving even more DOFs than the traditional two or three dimensional interfaces, are better integrated. The rest of this paper presents an application (though mediated

by a modern computer) of the thinking and analysis developed more than 40 years ago to data produced in a 6 DOF tracking task with integrated control and display.

THE EXPERIMENT

Experimental Task

The task involved pursuit tracking in 6 DOF. Subjects controlled a 3D cursor to align it as closely as possible in both position and angular orientation with a 3D target that moved unpredictably (Figure 1). Both the target and the cursor were tetrahedrons. To ensure that only one possible correct orientation match existed each tetrahedron had two blue adjacent edges; the remaining edges were colored red.

The cursor and the target had two differences to help minimize potential confusion: 1) the radius (center to any vertex) of the cursor was 1.3 times that of the target; 2) the cursor had semi-transparent surfaces while the target was a "wireframe" model.

The target motion was driven by six independent forcing functions with identical frequency characteristics, one for each DOF. These were weighted combinations of 20 sine functions with a random initial phase, such as:

$$x(t) = \sum_{i=0}^{19} A p^i \sin(2\pi f_0 p^i t + \phi_x(i)) \quad (1)$$

where t is the time duration from the beginning of each experimental test and the constants $A = 3.5$, $p = 1.25$, $f_0 = 0.01$. These values were set through pilot testing so that the target remained within the bounds of the display and moved at a challenging but manageable speed. $\Phi_x(i)$ ($i = 0, \dots, 19$) were independent pseudo-random numbers between 0 and 2π .

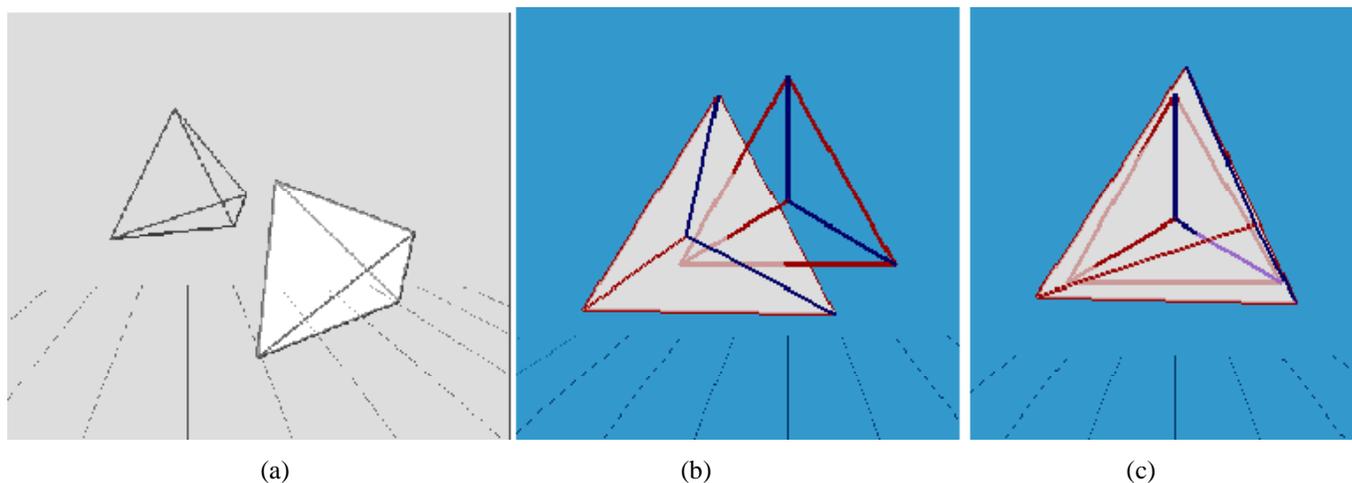


Figure 1. 6 DOF tracking task. The tetrahedron with semi-transparent surfaces is the cursor. The tetrahedron without semi-transparent surfaces is the randomly moving target. Subjects tried to align the cursor with the target. Shown in the figure are examples of (a) a very large 6 DOF error between cursor and target (b) a large translation error and small rotation error, and (c) a small translation error and large rotation error.

Experimental Apparatus

Display. In designing the 3D displays used in the experiment, four types of depth cues were chosen: binocular (stereoscopic) disparity, linear perspective, interposition (edge occlusion), and partial occlusion through semi-transparency. Binocular disparity, linear perspective and interposition are conventionally recognized as strong depth cues (Kaufman, 1974). The use of semi-transparency to create partial occlusion, as shown in Figure 1, is a novel technique, but has been shown to be both effective and easy to implement (Zhai, Buxton, & Milgram, 1996). During the experiment, subjects sat 60 cm away from the display and wore appropriate stereoscopic glasses. The experimental room was darkened throughout the experiment.

Input Controllers. Two 6 DOF input controllers were used in the experiment: a Spaceball™ and an Elastic General-purpose Grip controller (EGG), an egg shaped 6 DOF device designed by the first author (Figure 2). The Spaceball™ is an isometric, force sensitive device and the EGG is a suspended elastic resistance device whose displacement is proportional to the force and torque applied by the user. Both devices were operated in rate control mode.

Subjects. Thirty paid volunteers were screened. Three subjects were rejected for having weak stereoscopic acuity, and one was rejected for having poor corrected near-vision acuity. The accepted 26 subjects' ages ranged from 18 to 37. None of them had had prior experience with any 6 DOF manipulation devices. Thirteen subjects were assigned to the isometric rate controller (Spaceball™) and the remaining subjects to the EGG.

Procedure. Data gathering occurred over five phases. Each phase consisted of a practice session (3 minutes for Phase 1 and 7 minutes for the rest of the phases), followed by 4 trials of tracking. Each trial lasted 40 seconds. The entire experiment, including screening tests, lasted one hour for each subject.

TIME-ON-TARGET (TOT) BASED COORDINATION ANALYSIS

Method

A computer program, TOTscope, was developed to analyze TOT based coordination performance. TOT's in each of the 6 degrees of freedom were first calculated. The translational TOT's parameters were easy to compute. TOTx was defined as the total sum of time periods in which the translational distance between the target and the cursor in X dimension was smaller than a given threshold, divided by the total trial time (40 seconds). TOTy and TOTz were similarly defined.

TOTrx, TOTry, TOTrz, the time on target in rotational degrees of freedom required more careful definition. We used the X, Y, Z components of the rotation vector (See

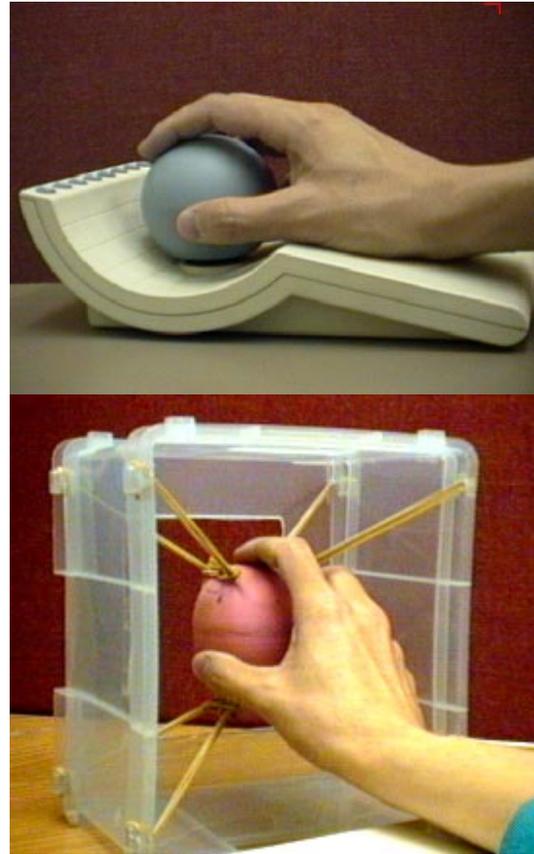


Figure 2. The isometric Spaceball™ (top) and elastic EGG (bottom) input controllers used in the experiment

Altmann, 1986) between the target and the cursor as the basis for calculating TOTrx, TOTry and TOTrz. TOTrx was defined as the total sum of time periods in which the X component of the rotation vector between the target and the cursor was smaller than a given threshold, divided by the total trial time (40 seconds). TOTry and TOTrz were similarly calculated.

Three higher order parameters, TOTb and TOTmin and STOT were then calculated. TOTb was the baseline target-on-target value, i.e. the probability purely by chance for all 6 degrees of freedom to be on target at the same time (Senders, 1956):

$$\text{TOTb} = \text{TOTx} \text{ TOTy} \text{ TOTz} \text{ TOTrx} \text{ TOTry} \text{ TOTrz} \quad (2)$$

TOTmin is simply the smallest of TOTx, TOTy, TOTz, TOTrx, TOTry, TOTrz, i.e.,

$$\text{TOTmin} = \min(\text{TOTx}, \text{TOTy}, \text{TOTz}, \text{TOTrx}, \text{TOTry}, \text{TOTrz}) \quad (3)$$

STOT is the actual percentage of time-on-target simultaneously in all 6 DOF. The final parameter that TOTscope seeks is coefficient C:

$$C = (STOT - TOTb)/(TOTmin - TOTb) \quad (4)$$

C was intended to reflect the quality of coordination. For $0 < C < 1$, $TOTb < STOT < TOTmin$, the trial is COORDINATED to the degree C indicates. (When $C = 1$, i.e. $STOT = TOTmin$, it is PERFECT coordination.) For $C = 0$, $STOT = TOTb$, it is no better than chance (UNCOORDINATED). For $C < 0$, $STOT < TOTb$, it is worse than chance (DISCOORDINATED).

Results

Figure 3 shows the TOT measures when 10, 20, 30 and 40 degrees of mismatch were chosen as the "on target" criterion. The thresholds for translational degrees of freedom were equivalent to the rotational threshold (i.e. rotational

threshold multiplied by the cursor radius). Note that each data point on the graph is the mean of 26 subjects, each of whom performed 4 trials in each experimental phase.

Upon inspecting the graphs, the following observations can be made:

1. Although not perfect, subjects' 6 DOF tracking trials are coordinated as measured by the C ratio for all four thresholds. For the 10 degree threshold, C was about 0.15 and for all other thresholds, C was approximately 0.2. Remember that C could range from -1, discoordinated, to 0, uncoordinated, to 1, (perfectly) coordinated.
2. TOTmin, STOT and TOTb all improved in later experimental phases but the C value remained almost constant. This was contrary to our expectation of seeing an increase of the C value, as subjects gained more experience.

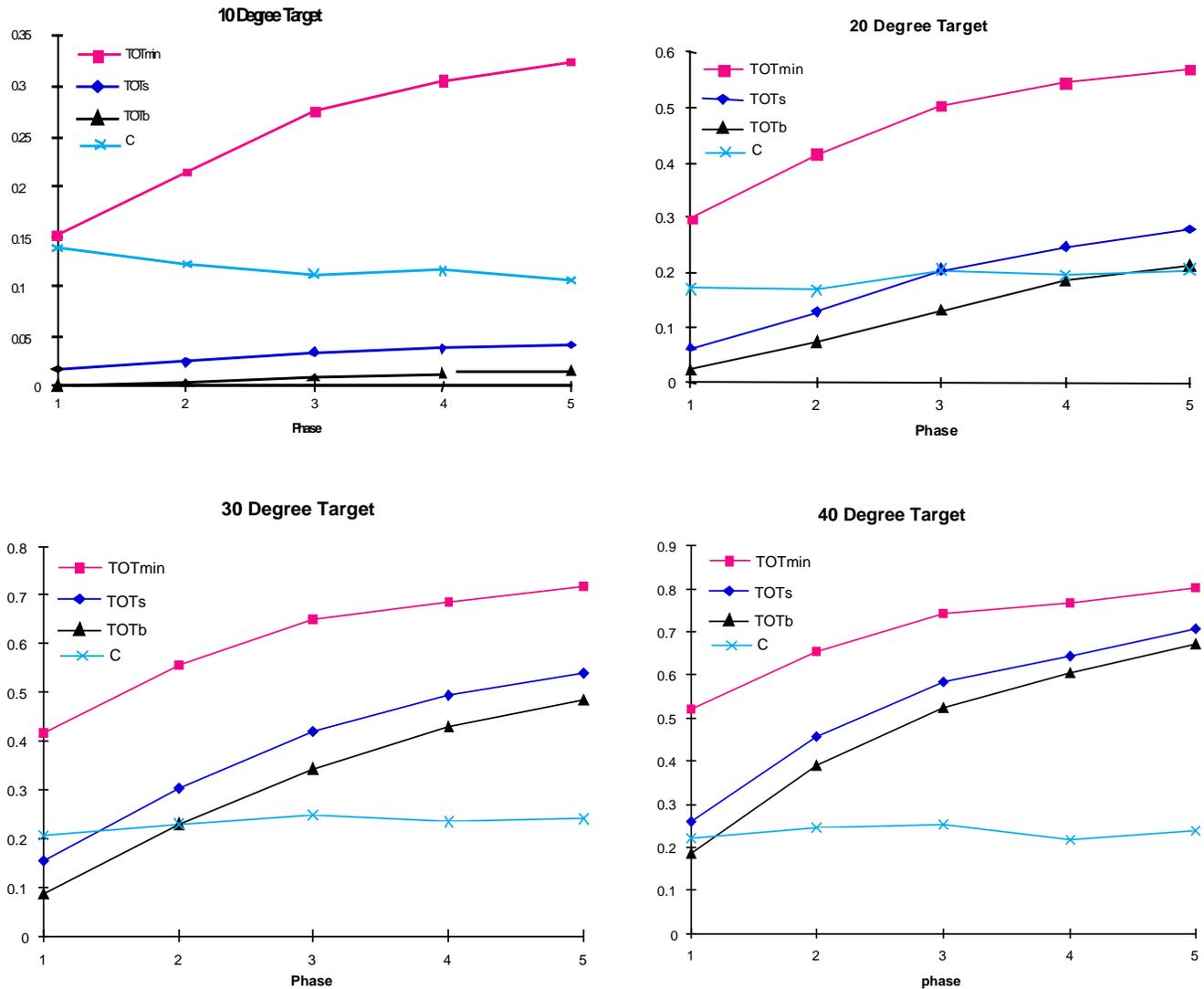


Figure 3. TOT measures under 10°, 20°, 30° and 40° time on target criteria

CONCLUSION / TRANSITION

The modern data are at variance with those found by Ellson in a 3 DOF task. Using a TOT based coordination measure, we found that even 6 degrees of freedom can be. It is our contention that the differences arose in part because of the differences in the design of the control and display spaces. In particular, the modern controller permits to a much higher degree than the PSMT non-interacting control outputs by the human controller using one hand. Both Ellson's and Senders' subjects used two hands in control and it may be that discoordination arise there as well. In the next part of this work we examine the linear correlations of the data.

ACKNOWLEDGEMENT

The experiment presented in the paper was conducted in the University of Toronto's Ergonomics in Teleoperation and Control (ETC) Laboratory directed by Professor Paul Milgram who has provided much support and advice on this study. We also gratefully acknowledge the funding support from the Information Technology Research Center (ITRC), a center of excellence of the province of Ontario.

References

- Altmann, S. L. (1986). *Rotations, Quaternions, and Double Groups*. Oxford: Clarendon Press.
- Brooks, T. L., & Bejczy, A. K. (1985). *Hand controllers for teleoperation, a state of the art technology survey and evaluation* (JPL Publication 85-11): Jet Propulsion Laboratory.
- Ellson, D. C. (1947). *The independence of tracking in two and three dimensions with the B-29 pedestal sight* (Memorandum Report TSEAA-694-2G): Aero Medical Laboratory.
- Gardner, J. F. (1950). *Direction of pointer motion in relation to the movement of flight controls* (Technical Report 6016): AMC.
- Jacobus, H. N., Riggs, A. J., Jacobus, C. J., & Weinstein, Y. (1992). Implementation issues for telerobotic handcontrollers: human-robot ergonomics. In M. Rahimi & W. Karwowski (Eds.), *Human-Robot Interaction*. London: Taylor & Francis.
- Kaufman, L. (1974). *Sight and mind - an introduction to visual perception*. London: Oxford University Press.
- O'Hara, J. (1987,). *Telerobotic control of a dextrous manipulator using master and six-DOF hand controllers for space assembly and servicing tasks*. Paper presented at the Human Factors Society 31st Annual Meeting, New York City.
- Rice, J. R., Yorchak, J. P., & Hartley, C. S. (1986,). *Capture of satellites having rotational motion*. Paper presented at the Human Factors Society 30th Annual Meeting, Dayton, Ohio.
- Senders, J. W., Christensen, J. M., & Sabeh, R. (1955). *Comparison of single operator's performance with team performance in a tracking task* (TN-55-362): Aero Medical Laboratory, Wright Air Development Center.
- Senders, J. W., Wallis, R., & Senders, V. L. (1956). *Coordination measures on two dimensional tracking tasks*. Unpublished manuscript.
- Zhai, S. (1995). *Human Performance in Six Degree of Freedom Input Control*. Ph.D. Thesis, University of Toronto.
[Http://vered.rose.toronto.edu/people/shumin.html](http://vered.rose.toronto.edu/people/shumin.html)
- Zhai, S., Buxton, W., & Milgram, P. (1996). The partial-occlusion effect: utilizing semi-transparency in 3D human-computer interaction. *ACM Transactions on Computer-Human Interaction*, 3(3), 254-284.

INVESTIGATING COORDINATION IN MULTIDEGREE OF FREEDOM CONTROL II: CORRELATION ANALYSIS IN 6 DOF TRACKING

Shumin Zhai
IBM Almaden Research Center
San Jose, California, USA

John W. Senders
University of Toronto
Toronto, Ontario, Canada

In these two companion papers, methods developed in a series of studies in the 1940's and 1950's are applied to the analysis of 6 DOF control devices used in modern human machine systems such as teleoperation and virtual environments. Contrary to the early studies, the current work showed that the simultaneous time on target in multidegree of freedom tracking was higher than the product of component time on target scores. The distribution of linear correlation coefficients between the tracking errors of different degrees of freedom tended to be skewed towards the positive values. These results suggested that subjects' discoordination in early multidegree of freedom tracking studies was likely due to the limitation of human machine interfaces at that time. With well designed interfaces, subjects exhibited more coordinated trials than discoordinated trials in multidegree of freedom tracking.

INTRODUCTION

The companion paper "Investigating Coordination in Multidegree of Freedom Control I: Time-on-Target Analysis of 6 DOF Tracking" introduced the historical background of coordination in multidegree of freedom control, described a 6 DOF tracking experiment and presented a Time-on-Target (TOT) based analysis of coordination performance in the experiment. The TOT based analysis showed a trend opposite from that found in the early studies: simultaneous TOT (STOT) tends to be greater than the product of TOTs in the component dimensions, suggesting the existence of coordinated multiple degree of freedom control.

Complementary to TOT based coordination analysis, we have also conducted a linear correlation analysis based on the same 6 DOF tracking experiment described in Part I. We computed the correlation coefficient, r , directly from the tracking errors of the various degrees of freedom. A positive linear correlation indicates the degree that two variables co-vary. Its magnitude indicates the degree that the two errors are simultaneously reduced. In this sense, a correlation coefficient can serve as a measure of coordination, as suggested in (Senders, Wallis, & Senders, 1956) in their use of the Phi coefficient (derived from the Product Moment Correlation).

In addition to correlations between different pairs of the 6 degrees of freedom, we also calculated correlations between the Euclidean total magnitude of translation mismatch and total rotation mismatch (magnitude of the rotation vector) in each trial of tracking in the experiment. Our goal was to gain insights as to whether translation and rotation were integrated or separated aspects of 3D object manipulation. As reviewed in Part I of this paper, a persistent controversy in 6 DOF manipulation has been whether rotation and translation should

be assigned to two separate hands (Rice, Yorchak, & Hartley, 1986, O'Hara, 1987, McKinnon & King, 1988). In lights of Jacob and colleagues (Jacob, Sibert, McFarlane, & Mullen, 1994), only those variables that are perceptually integrated should be controlled with one multidegree of freedom controller. Similar conclusions arise from the proximity compatibility principle (Wickens, 1992).

RESULTS

The Distribution of Correlation Coefficients

The correlation coefficient (r) between every two degrees of freedom, such as X-Y, X-Z, Y-Z, in each trial of 6 DOF tracking was calculated. Figure 1 shows the distribution of r from 26 subjects, 5 experimental phases, 4 tracking paths and two types of controllers all lumped into one plot. Approximately 3/4 of the r 's were on the positive side ($r > 0$)

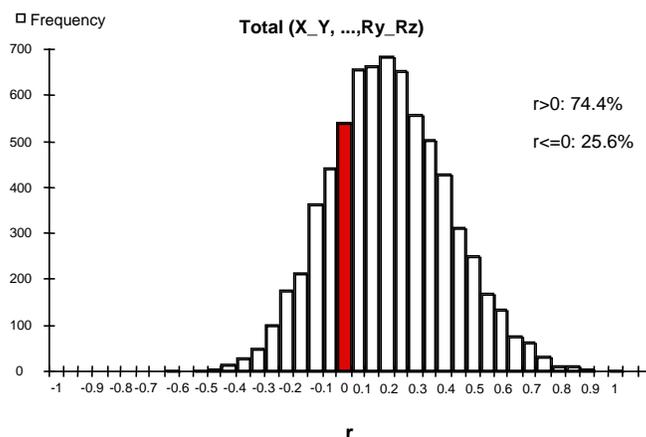


Figure 1. Total distribution of correlation coefficient r

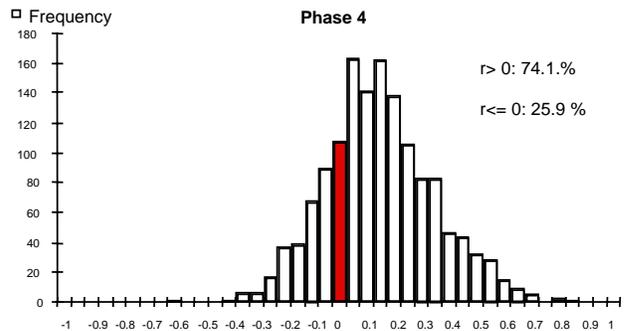
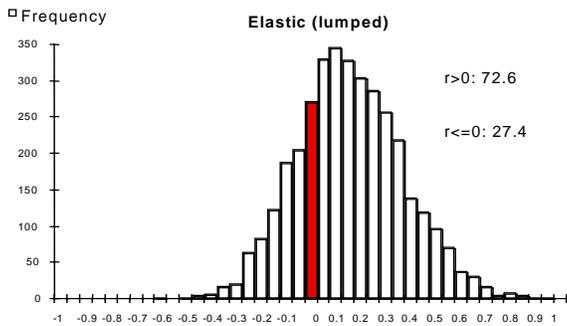
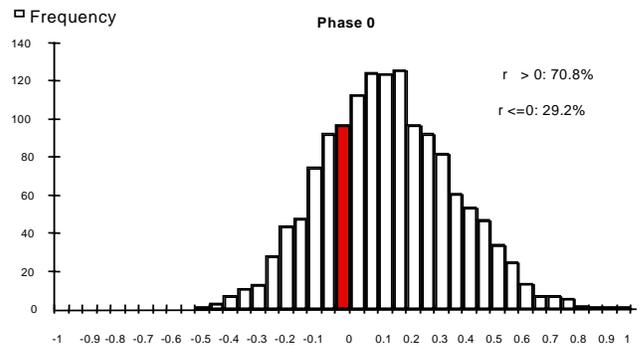
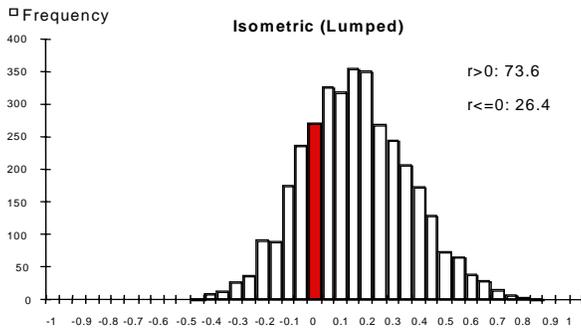


Figure 2 Similar distribution of correlation coefficient was found for two different controllers

Figure 3 The distribution of correlation coefficients were consistent across experimental phase

and 1/4 were zero or negative ($r \leq 0$). In other words, there were more coordinated pairs of degrees of freedom than discoordinated ones.

The Consistency of the r Distribution

The r distribution varied little from controller to controller (Figure 2), from the first experimental phase to the last experimental phase (Figure 3) or from one tracking trajectory to another. It was also very consistent for all pairs of degrees of freedom, such as between X and Y, or between X and R_y , or between R_x and R_y (Figure 4). There was not one pair of degrees of freedom that was particularly more discoordinated than any other pair.

Coordination Between Translation and Rotation

The positive valued correlation between total translational error magnitude (Euclidean distance) and the total rotation error were more frequent than those between all pairs of DOF, with 88.6% trials positively correlated and 11.4% negatively correlated (Figure 5). This suggests that the rotational and translational aspects of the 3D tracking task were treated in an integrated manner by the subjects.

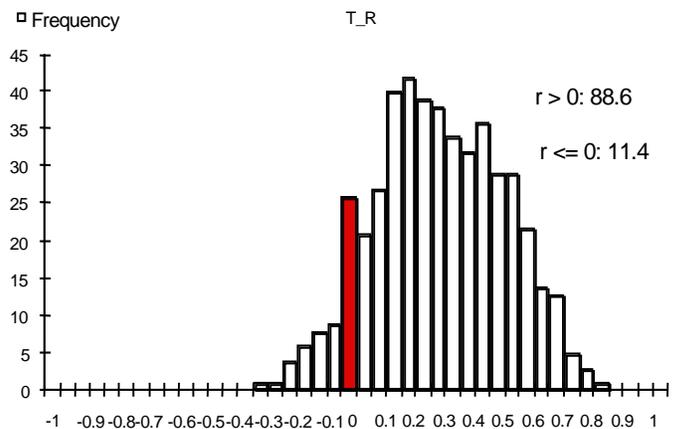


Figure 5 High degree of correlation coefficients were found between translation and rotation

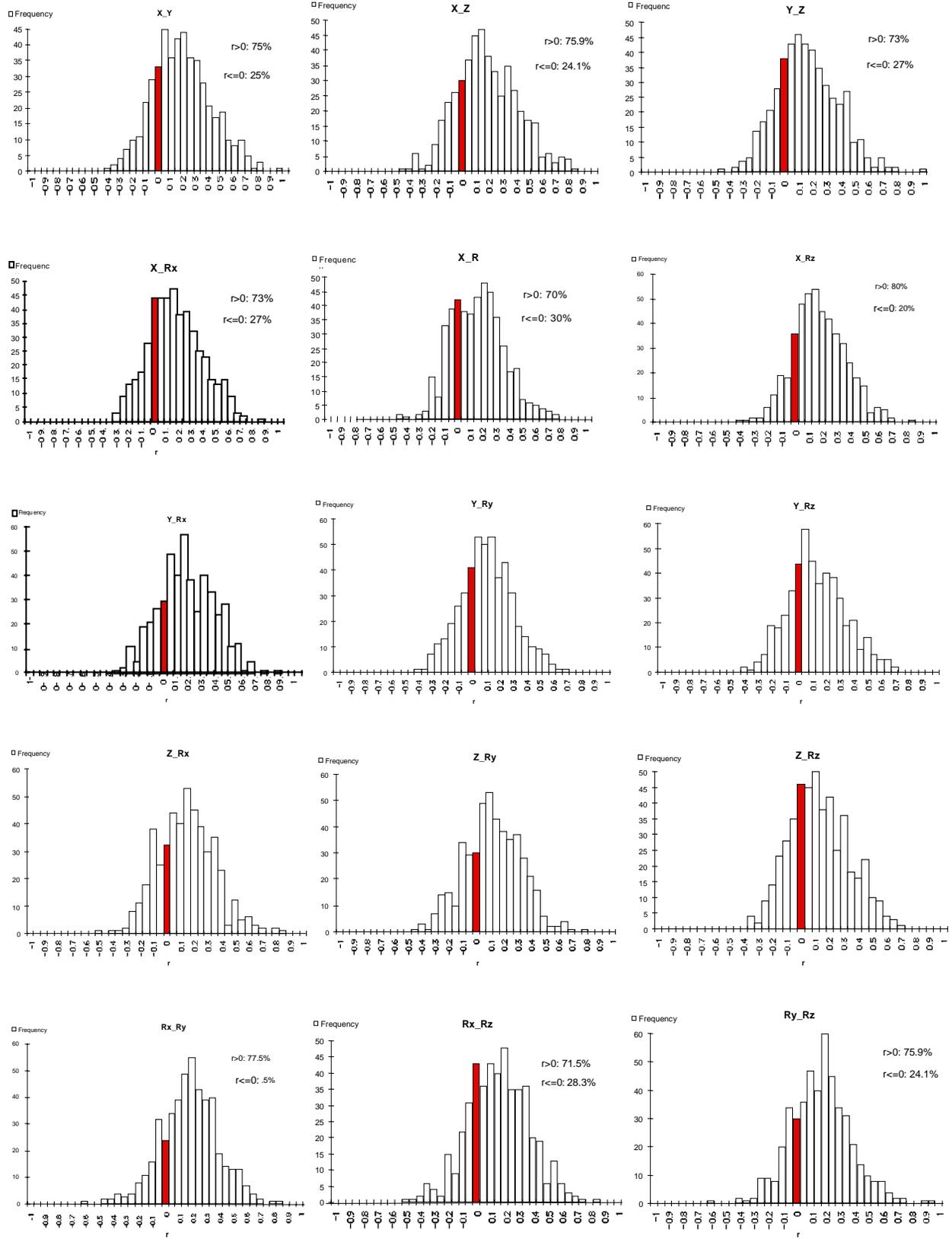


Figure 4 The distribution of correlation coefficients were consistent across different pairs of degrees of freedom

Individual Differences

There were individual differences in the distribution of correlation coefficient r . Figure 6 shows a sample of individual data. For Subject C, only 59.7 % tracking was positively correlated, but the mean r value was 0.18. For Subject I, 85.7% tracking was positively correlated, but the mean r value was only 0.12. Subject K had 66.7% positively correlated tracking, with mean r value at 0.20.

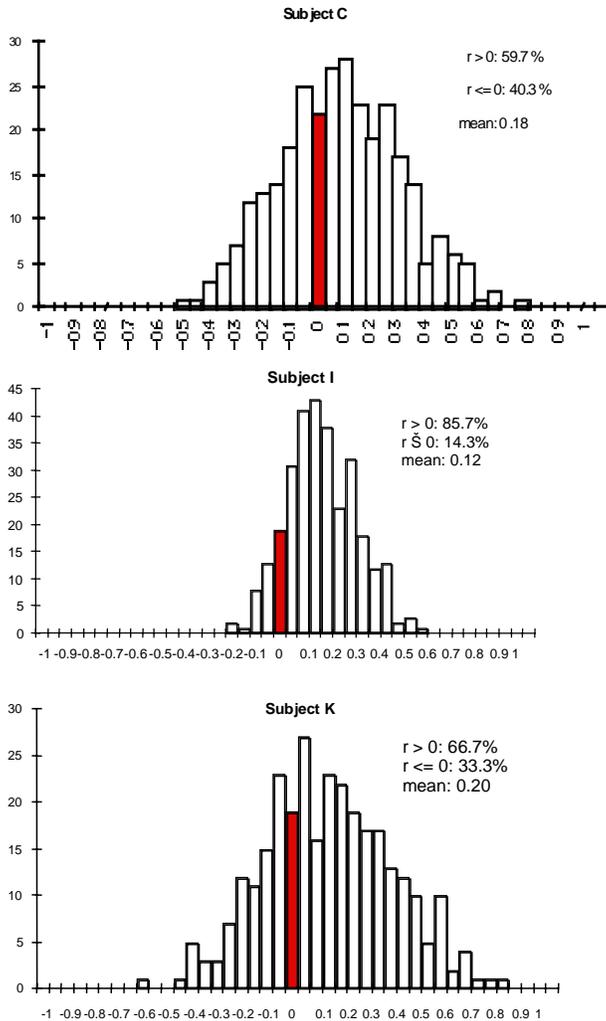


Figure 6 A sample of individual r distribution

CONCLUSIONS AND DISCUSSIONS

Contrary to early findings with regards to multiple DOF tracking, the present study revealed that subjects *could* coordinate multiple degrees of freedom, if the human machine interfaces are designed properly. Early studies (Ellson, 1947); (Gardner, 1950; Senders, Christensen, & Sabeh, 1955; Senders et al., 1956) showed that in 2 or 3 DOF tracking tasks, subjects' simultaneous TOT tended to be lower than the

product of component TOT's in each dimension, suggesting while tracking error in one dimension was reduced, errors in other dimensions tended to increase. The current study showed that in a tracking task with more degrees of freedom (6), simultaneous TOT tended to be higher than the product of the component TOT's.

The distribution of linear correlation coefficient in the present study was, interestingly, a mirror image of what was found earlier (Compare Figure 1 with Figure 7). Figure 1 shows that about 3/4 of the tracking trials were positively correlated while Figure 7 shows about 3/4 of the trials to be negatively correlated. Note again that the data in Figure 7 were collected from subjects dealing with two (or one pair of) degrees of freedom while data in Figure 1 were collected from subjects controlling six (or 15 pairs of) degrees of freedom.

If we examine the task at a higher level by taking total translational error and total rotational error as two variables, the correlation coefficients were even more positively distributed (Figure 5). This suggests that subjects treated rotation and translation as integrated aspects of a 3D object, providing evidence to support one handed design of 3D object DOF manipulation.

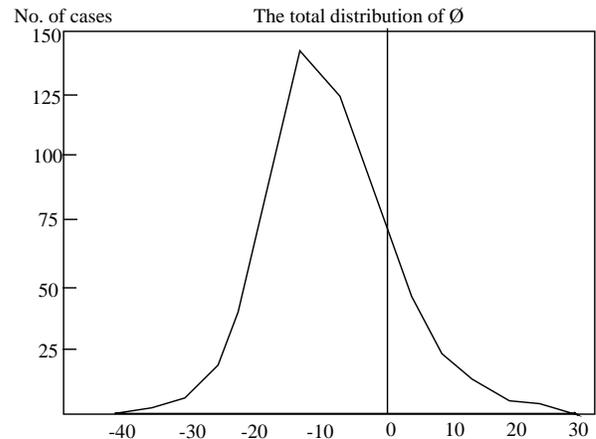


Figure 7 Senders' \emptyset distribution (Adapted from (Senders et al., 1956) Reprinted with permission)

We have taken two complementary approaches to the study of coordination in multidegree of freedom manual control. Both of the two approaches were motivated by early historical thinking. For the current analysis of 6 DOF tracking, the two approaches differed both computationally and conceptually. The TOT analysis was based on the comparison of simultaneous time-on-target in all 6 degrees of freedom and the product of 6 one DOF time-on-target in each dimension. In other words, it examined subjects' coordination in 6 DOF tracking at a higher level: if all but one degree of freedom was off target, it was not considered coordinated behavior. In contrast, the correlation analysis was conducted at a component level: it examined relationships between each and every pair of degrees of freedom.

Although the two approaches were different, the conclusions drawn from them are very consistent: there was little fundamental conflict in controlling 6 degrees of freedom, effort paid to one degree of freedom does not necessarily cause destructive interference to other degrees of freedom. On the other hand, neither the C factors derived from TOT's nor the means of the r distribution were very high, suggesting that subjects 6 DOF control was far from perfectly coordinated.

To conclude, the current study further develops methods in early multidegree of freedom tracking studies and links them to the analysis of modern 6 DOF control devices. Our results suggested that subjects' discoordination in early studies was likely due to the design and construction limits of human machine interfaces at that time. With the current interfaces, subjects exhibited more coordinated trials than discoordinated trials in multidegree of freedom tracking tasks.

In addition to one handed control in the current study vs. two handed control in Ellson's and Senders' studies, there are a few other interface differences as well. The display space and the control space in the current study are much more isomorphic than in the earlier work. Also the control space is more perfectly harmonized. By this it is meant that the control output in display terms, the C/D gain, was uniform across all inputs for the isometric control and optimized for subjective harmonization for the EGG. Furthermore, the current 6 DOF interfaces use rate control instead of position control as in the early studies. It can be speculated that rate control may better facilitate coordination, since it removes the anatomical constraints of the human arm. Any or all of these differences may have contributed to the shift from discoordination to coordination. We must also point out that the subjects in the present study have grown up in a different environment from those of the 1940s and 1950s. Most important is the question of exposure to computer games.

Remaining Issues and Future Work

Although measures used in this study have been informative on the control coordination issues that we are interested in, they are far from ideal. First, complete linear cross-correlation functions (with various phase lags), instead of just correlation coefficients with zero lag, should be derived from the experimental data. If subjects still had to quickly switch attention and control between degrees of freedom, there would be a peak correlation between different degrees of freedom at non-zero phase lag. Senders did perform such an analysis for one trial of one subject and found that the correlation, negative at $\tau = 0$, became positive at a time shift of about .45 seconds (Senders, 1996). The difficulty in such an analysis lies in its computational demands. A function, not a coefficient, has to be computed between every pair of degrees of freedom for every trial of tracking data. These functions then have to be aggregated in a mathematically valid approach. Future work also includes seeking alternative measures of coordination (Zhai, 1995).

ACKNOWLEDGEMENT

The experiment presented in the paper was conducted in the University of Toronto's Ergonomics in Teleoperation and Control (ETC) Laboratory directed by Professor Paul Milgram who has provided much support and advice on this study. We also gratefully acknowledge the funding support from the Information Technology Research Center (ITRC), a center of excellence of the province of Ontario.

References

- Ellson, D. C. (1947). *The independence of tracking in two and three dimensions with the B-29 pedestal sight*. Memorandum Report TSEAA-694-2G, Aero Medical Laboratory.
- Gardner, J. F. (1950). *Direction of pointer motion in relation to the movement of flight controls* (Technical Report 6016): AMC.
- Jacob, R. J. K., Sibert, L. E., McFarlane, D. C., & Mullen, M. P. (1994). Integrality and separability of input devices. *ACM Transactions on Computer-Human Interaction*, 1(1), 3-26.
- McKinnon, M., & King, M. (1988,). *Manual control of telemanipulators*. Paper presented at the International Symposium Teleoperation and Control, Bristol, England.
- O'Hara, J. (1987,). *Telerobotic control of a dextrous manipulator using master and six-DOF hand controllers for space assembly and servicing tasks*. Paper presented at the Human Factors Society 31st Annual Meeting, New York City.
- Rice, J. R., Yorchak, J. P., & Hartley, C. S. (1986,). *Capture of satellites having rotational motion*. Paper presented at the Human Factors Society 30th Annual Meeting, Dayton, Ohio.
- Senders, J. W., Christensen, J. M., & Sabeh, R. (1955). *Comparison of single operator's performance with team performance in a tracking task* (TN-55-362): Aero Medical Laboratory, Wright Air Development Center.
- Senders, J. W., Wallis, R., & Senders, V. L. (1956). *Coordination measures on two dimensional tracking tasks*. Unpublished manuscript.
- Wickens, C. D. (1992). *Engineering Psychology and Human Performance*: HarperCollines Publishers.
- Zhai, S. (1995). *Human Performance in Six Degree of Freedom Input Control*. Ph.D. Thesis, University of Toronto.
[Http://vered.rose.toronto.edu/people/shumin.html](http://vered.rose.toronto.edu/people/shumin.html)

Quantifying Coordination in Multiple DOF Movement and Its Application to Evaluating 6 DOF Input Devices

Shumin Zhai

IBM Almaden Research Center
650 Harry Road, San Jose, CA 95123, USA
+1 408 927 1112 zhai@almaden.ibm.com

Paul Milgram

Dept Mech. & Ind. Eng., University of Toronto
Toronto, Ontario, Canada, M5S 3G8
+1 416 978 3662 milgram@mie.utoronto.ca

ABSTRACT

Study of computer input devices has primarily focused on trial completion time and target acquisition errors. To deepen our understanding of input devices, particularly those with high degrees of freedom (DOF), this paper explores device influence on the user's ability to coordinate controlled movements in a 3D interface. After reviewing various existing methods, a new measure of quantifying coordination in multiple degrees of freedom, based on movement efficiency, is proposed and applied to the evaluation of two 6 DOF devices: a free-moving position-control device and a desk-top rate-controlled hand controller. Results showed that while the users of the free moving device had shorter completion time than the users of an elastic rate controller, their movement trajectories were less coordinated. These new findings should better inform system designers on development and selection of input devices. Issues such as mental rotation and isomorphism vs. tools operation as means of computer input are also discussed.

Keywords

Input devices, interaction techniques, evaluation methods, 6 DOF control, rotation, mental rotation, 3D interfaces, virtual environments, motor control, coordination.

INTRODUCTION

Computer input control has traditionally been evaluated using speed (e.g., task completion time) or accuracy (e.g., error rate) as performance measures. As we move to broader topics such as drawing [2], two handed input [8] and high degree-of-freedom (DOF) control, these measures become insufficient to capture the complete quality of input performance.

Driven by the need in 3D user interfaces, much research has been done to evaluate various multiple DOF input devices [see 6, 7, 15, 14 for reviews]. Many fundamental questions on multi-DOF input, however, remain to be scientifically addressed. Can users simultaneously control all 6 degrees of freedom? Or do users actually control

fewer degrees of freedom at a time? Can one 6 DOF device be substituted with multiple lower DOF devices? Rice et al [12] observed that controlling 6 DOF with one hand is difficult. Some teleoperation systems, such as the Shuttle Remote Manipulator, also known as the "Canadarm", require two-handed operation: one hand for rotation control and the other for translation control. O'Hara [9] contradicted such an observation and found little performance difference between two 3 DOF controllers and one 6 DOF controller. To answer these questions on a firm scientific ground, we first need to define informative measures beyond speed and accuracy. One of them is the degree of coordination among the multiple degrees of freedom.

COORDINATION MEASURES

For a given trial of motor performance, such as an athlete's movement or a trial of docking in 3D space, people can often agree if it is coordinated. The research challenge here is how to reflect consensual and intuitive understanding by quantitative measures. In the case of multiple degrees of freedom input control, the following measures have been considered as indices of coordination.

Simultaneity. For a task that involves multiple degrees of freedom, coordinated control may require all the degrees of freedom simultaneously activated. Percentage duration that multiple degrees of freedom are co-activated can therefore be a measure of coordination. The drawback of the simultaneity measure is that it does not account for the magnitude of the control actions in each degree of freedom. As long as all of the degrees of freedom are activated, regardless the amount of input generated, the trial is considered coordinated by this measure.

Time-on-target and correlation. In a 3 DOF pursuit tracking task, Ellson [3] recorded *simultaneous* time-on-target (STOT) in all pairs of degrees of freedom as well as all three at once, in addition to TOT (time-on-target) in each of the component dimensions. He then compared STOT scores with the *products* of the component TOT scores. His argument was that if the percent STOT was *equal* to the product of the component TOTs, then the components may be considered independent (uncoordinated). If greater, they were positively correlated (coordinated); if less, negatively correlated

(discoordinated). Senders [13] extended Ellson's approach. He computed an approximation of the product-moment correlation coefficient of two separate degrees of freedom as a measure of coordination. Recently, Zhai and Senders [17, 18] extended the time-on-target and correlation measures to 6 DOF tracking tasks and found that subjects tended to have coordinated trials when using a 6 DOF isometric or elastic rate control device. However, both TOT and correlation as coordination measures have drawbacks. One of them is that these two measures do not account for perfect trials. If a trial is 100% simultaneously on target with zero tracking error in all degrees of freedom at all time, the TOT coordination measure will give an uncoordinated result (STOT equals to the product of component TOTs) and correlation become meaningless since errors in all degrees of freedom are zero.

EFFICIENCY AS COORDINATION MEASURE

We propose *efficiency* as a measure of quantifying coordination in multiple degrees of freedom. For a task that involves N degrees of freedom, the trajectory that has the shortest length in that N dimensional space is considered the most coordinated movement. For simplicity, let us examine trajectories on a 2D space, as illustrated in Fig. 1. In order to move from Point A to B in this space, two variables x and y have to be changed from x_A to x_B and y_A to y_B respectively. Supposing we had an input device that has two separate 1 DOF controls, as in an Etch-a-Sketch toy, one possible trajectory would be AC-CB, as a result of moving in the x dimension first and in the y dimension second. In such a case, the two degrees of freedom are completely uncoordinated, because x and y are not moved at the same time, resulting in a longer trajectory than necessary. With an integrated 2 DOF device such as a mouse, one may produce a trajectory *l* that is close to the straight line AB. Trajectory AB is the shortest and most efficient among all possible trajectories. It can also be considered most coordinated in the sense that x and y move simultaneously at the *same relative pace*. Any deviation from the path AB can be considered a result of imperfect coordination, which will result in a longer trajectory. In light of this analysis, we define the translation inefficiency, i.e. the amount of "work wasted", as an inverse measure of translation coordination.

$$\frac{\text{Length of actual path} - \text{Length of shortest path}}{\text{Length of shortest path}} \quad (1)$$

By this definition, trajectory *l* in Fig. 1 is better coordinated than trajectory AD-DB, which is in turn better coordinated than AC-CB.

The same definition of coordination coefficient can be easily generalized to translations in 3D space simply by measuring 3D instead of 2D Euclidean distances. Fig. 2 shows (top curves) an example of the 3D application.

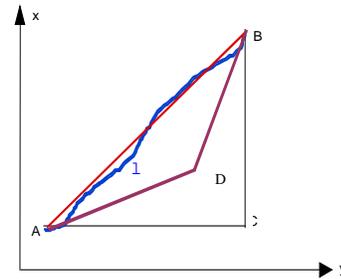


Fig. 1: Coordination with 2 degrees of freedom

To generalize the coordination measure to rotation in 3D space is less straightforward, however. The parameters commonly used in engineering (Euler angles), pitch, yaw, roll, are often misleading [1]. A more valid metric is *rotation vector*. Define the initial mismatch between a cursor and a target (both are 3D objects in 3D space) to be

$$\mathbf{\Theta}_A = \phi_A \mathbf{n}_A = (\phi_{Ax}, \phi_{Ay}, \phi_{Az}) \quad (2)$$

where $\mathbf{\Theta}_A$ is the rotation vector signifying an angle ϕ_A of

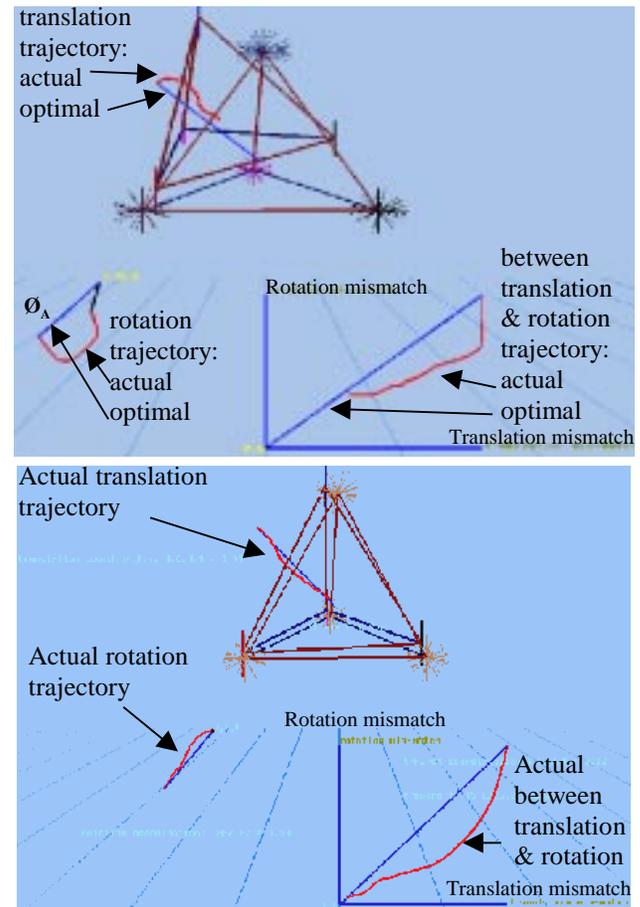


Fig. 2. Coordination measurements superimposed on to a 6DOF docking task. Top: a trial in progress. Bottom: a completed trial. The ratio between actual and optimal trajectory in translation(3D), rotation(3D), and between translation-rotation spaces(2D) quantify the degree of coordination.

rotation about \mathbf{n}_A , where $\mathbf{n}_A = (n_x, n_y, n_z)$ is a unit vector defining the axis of the rotation in x, y, z frame of reference, then the *minimum* amount of rotation that the cursor has to go through to reach the target is ϕ_A (Fig. 2). The ratio between ϕ_A and the actual amount of rotation of the cursor upon reaching the target is defined as the rotation coordination coefficient:

$$\frac{\text{Amount of actual rotation} - \text{Initial rotation mismatch}}{\text{Amount of actual rotation}} \quad (3)$$

When one can control all 3 rotational degrees of freedom with *perfect* coordination, the rotation mismatch between the cursor and the target will be reduced from $\phi_A \mathbf{n}_A$ to $0 \mathbf{n}_A$, *without changing the direction of the mismatching rotation vector*. Otherwise, if the 3 rotational degrees of freedom cannot be controlled *simultaneously at the same relative pace*, at an instant of time t the mismatch will be $\phi_t \mathbf{n}_t$ ($\mathbf{n}_t \neq \mathbf{n}_A$), causing a larger amount of actual cursor rotation (Fig. 2, actual trajectory).

The two coordination coefficients defined above deal with translation and rotation separately but do not reveal the coordination aspect between translation and rotation taken together. In other words, a trial can be perfectly coordinated with respect to both its translation trajectory and rotation trajectory, but the rotation and the translation may not necessarily be performed at the same time. Hence, a third coordination factor is defined in the translation-rotation (2D) space which has two dimensions. One is the translation distance, d_t , between the cursor and the target centers of mass, and the other is the rotation mismatch ϕ_t (the magnitude of rotation vector) between the cursor and the target (See Fig. 2). Note that both d_t and ϕ_t are function of time, which define a 2D trajectory over the course of an experiment trial.

Defining coordination by optimality in fact has been proposed in the human motor control literature. For example, Flash and Hogan have measured coordination by the minimum jerk (rate of change of acceleration) for arm movement [4].

COORDINATION IN TWO CLASSES OF 6 DOF INPUT DEVICES: AN EXPERIMENT

Having defined the efficiency measure of coordination, we applied it in two experiments to investigate two 6DOF input devices, the Fingerball and the EGG¹ (Fig. 3).

The Fingerball, similar in shape and size to the 3Ball of Polhemus, represents a class of isotonic (free moving) 6

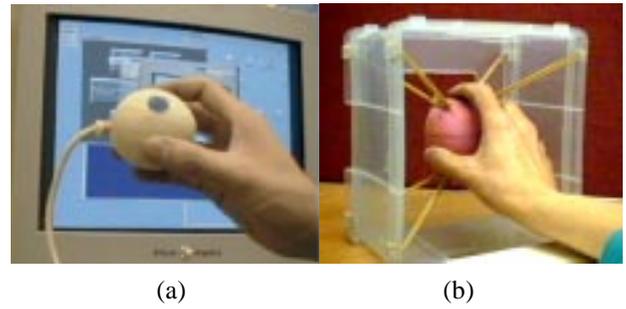


Fig. 3. Two 6 DOF devices used in the experiments. (a) The Fingerball is a free moving, position controlled input device and (b) The EGG is a desktop, elastic rate controlled device. The two experimental devices were based on the same 6 DOF magnetic sensor, Ascension BirdTM.

DOF input devices. Previous study showed that the Fingerball is superior to glove-based devices, due to the fact that one can use both the fingers and the arm/wrist to manipulate the degrees of freedom [16]. The EGG (Elastic General-purpose Grip) represents a different class of 6 DOF devices that are constrained on desktop and work in rate control. In comparison to the commonly used isometric rate controlled desktop device such as the SpaceballTM, the EGG offers a slight advantage at the early learning stage due to richer proprioception [15].



Fig. 4. Experimental Set-up

Experimental Set-up

Experimental Platform and Display

The experiment was conducted with a desktop 3D virtual environment. In order to ensure that the task performance was driven predominantly by differences in input controller conditions rather than by difficulties in perceiving depth information, binocular depth cue was implemented by means of a 120 Hz sequential switching stereoscopic display, together with perspective projection and occlusion (Fig. 2, 4).

Experimental Task

A 6 DOF docking task, illustrated in Fig. 2 and 5, was used for this experiment. (The coordination displays in Fig. 2, superimposed onto the docking task, were not visible to the subjects). In the experiments, subjects were asked to move a 3D cursor as quickly as possible to align it with a 3D target. The cursor and the target were two

¹ The experiments also included an isometric 6 DOF device (the SpaceballTM). Due to space limit, this paper only analyzes and reports results with regard to the Fingerball versus the EGG.

tetrahedra of equal size (4.2 cm from the center to each vertex). The edges and vertex markers (bars and spherical stars) of both tetrahedra were colored so that there was only one correct match in orientation. The stars on the target indicated the acceptable error tolerance for each vertex (0.84 cm). During the trial, whenever a corner of the cursor entered into the tolerance volume surrounding the *corresponding* corner of the target, the star on that corner changed its color as an indication of capture. Whenever all four corresponding corners stayed concurrently matched for 0.7 seconds, the trial was deemed completed. At the end of each trial, the trial completion time was printed on the screen. The beginning of each trial was signaled with a long auditory beep and the end of each trial was signaled with a short beep.

At the beginning of each trial, the cursor appeared in the center of the 3D space while the target randomly appeared in one of 8 pre-set locations and orientations. The 8 trials were divided into two sets of 4 trials. In one set of the trials the cursor and the target were mismatched in orientation about axes that were parallel with the viewer's primary coordinates (X, Y, Z). In another set the orientation mismatches were about arbitrary vectors that did not coincide with the X, Y, Z coordinates. Recent research in mental rotation [10] has shown that humans cannot effectively perform mental rotation about arbitrary 3D axes. We hypothesized that once interaction (manipulation) is allowed, subjects should be able to find the correct rotation path. Note that the magnitude of mismatch of both translation and rotation in each trial in one set correspond to a trial in the other set, so the total amount of translation and rotation are equalized in the two groups.

Experimental Design

Both experiments used between-subject design in order to avoid asymmetrical skill transfer between devices [11]. In Experiment 1, each device group had 16 subjects, none of them had prior experience with using 6 DOF input devices.

Each experiment had 5 repeated tests, which consisted of randomly shuffled 8 trials (with initial locations and orientations as described earlier). Test 1 started after a

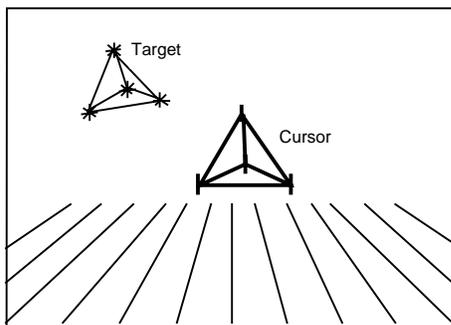


Fig. 5. 6 DOF Docking Task

short demonstration and two warm-up trials. Test 2, Test 3, Test 4, and Test 5 started 10, 20, 30, and 40 minutes after the beginning of Test 1 respectively. Practice trials (with completely random initial locations and orientation) were given between the tests. The entire experiment lasted about 1 hour for each subject.

A follow-up experiment, Experiment 2, was conducted for two reasons. First, we wanted to see effect of retention and extended practice. Second and more importantly, we wanted to know if a coordination difference between the two devices still exists if we give explicit instructions emphasizing coordination.

16 subjects (8 in each device group) who participated in Experiment 1 were called back two months later in Experiment 2, which started with the same instruction as in Experiment 1. After regaining their skills in Test 1 and Test 2, subjects were instructed (through demonstration and explanation) to perform the trials as coordinated (producing smooth and short trajectory) as possible, while trying to complete each trial as quickly as possible.

As a motivating tactic, before Test 3 of Experiment 2, completion times were displayed to the subjects after each trial and each test. After Test 2 of Experiment 2, these were displayed together with coordination measures.

Experimental Results and Discussion

The results of statistical analyses of data collected in the two experiments are summarized in Table 1.

Completion time. As shown in Fig. 6, for both experiments, the mean trial completion time of the free position control (Fingerball) group was significantly shorter than that of the elastic rate control (EGG) group. (Due to space constraint, all F-test degrees of freedom and significance level are summarized in Table 1).

Particularly worth noting is that after Test 2 in Experiment 2 when the emphasis on coordination was given, the subjects sacrificed their completion time in order to make more coordinated movements (Test 3). As they gained more practice, however, the completion time continued to improve.

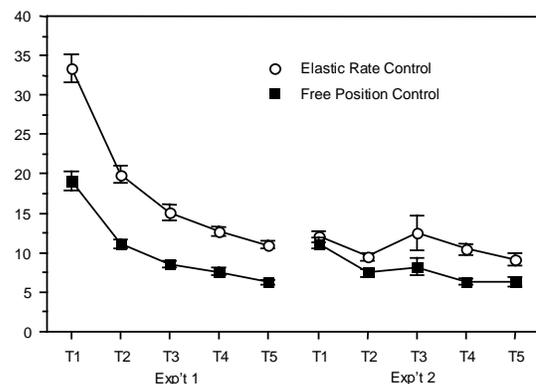


Fig. 6. Mean trial completion time with standard error bars

Table 1: Summary of Experiment 1 Variance Analyses

Independent variable	Device		Exp't Phase		Device X Phase		Rotation Type		RTYPE X Device		
	F1,30	P <	F4, 120	P <	F 4,120	P <	F1,30	P <	F1,30	P <	
Exp't 1	Completion Time	30.8	.0001	232.8	.0001	.116	.97 NS	27.6	.0001	.96	.33 NS
	Translation Inefficiency	44.4	.0001	56.4	.0001	1.7	.15 NS	35.1	.0001	.26	.61 NS
	Rotation Inefficiency	24.7	.0001	68.2	.0001	.197	.93 NS	58.0	.0001	16.2	.001
	Between Tran & Rot	18.0	.0005	94.5	.0001	1.20	.32 NS	13.1	.005	.308	.58 NS
	Total Transport	18.7	.0005	83.2	.0001	1.78	.14 NS	39.4	.0001	.772	.39 NS
Expt 2	Completion Time	5.31	.05	13.8	.0001	2.54	.06 NS	26.8	.0001	14.3	.005
	Translation Inefficiency	36.3	.0001	42.7	.0001	1.30	.27 NS	3.99	.07 NS	3.17	.10 NS
	Rotation Inefficiency	27.1	.0001	33.5	.0001	2.84	.05	40.8	.0001	38.2	.0001
	Between Tran & Rot	14.7	.005	5.58	.0001	4.42	.05	2.18	.16 NS	6.72	.05
	Total Transport	22.7	.0005	54.2	.0001	2.82	.05	24.8	.0005	13.9	.005

Translation. Fig. 7 illustrates the mean translation inefficiency measured in the experiments. In contrast to the trial completion time data, for both experiments, the free moving position control device was significantly (Table 1) less efficient than the elastic rate control device.

Subjects significantly improved their translation coordination over the five tests in each experiment, particularly after Test 2 of Experiment 2 when emphasis on coordination and efficiency was given. In terms of magnitude, on average the initial translation trajectories were 300% (free position control group) or 200% (elastic rate control group) longer than the optimal path. At the end of Experiment 2, the mean inefficiency of the elastic rate control group was reduced to 43.3% but that of the free position control group was still at 88.7%.

The lesser degree of coordination of the free moving position control device is plausible. First, position control is directly proportional to hand/finger movement and thus constrained to anatomical limitations: joints can only rotate to certain angle. In contrast, with an elastic rate control device, a small amount of hand movement is mapped onto the velocity of the cursor movement. The integral transformation (from velocity to cursor position) in rate control makes the actual cursor movement a step removed from the hand anatomy.

producing smoother trajectories than position control. This also contributes to the higher efficiency of rate control.

Rotation. Fig. 8 illustrates the mean rotation inefficiency of the two devices. Similar to translation inefficiency, for both experiments, the rotational inefficiency with the free position control device was significantly higher than with the elastic rate control device (Table 1). Subjects also significantly improved their rotation performance over the five tests in each experiment. At the end of Experiment 2, the mean rotation inefficiency of the free position group was reduced to 97.3% and elastic rate control group reduced to 70.4%.

Note that subjects' rotation inefficiency was much higher than that of translation, up to 580% in Test 1 of Experiment 1 by the free position control group. One possible reason is that humans can not effectively do mental rotation of 3D objects. In other words, subject might not be able to figure out the ideal rotation axis before they manually trying out the movement. We will return to this issue shortly.

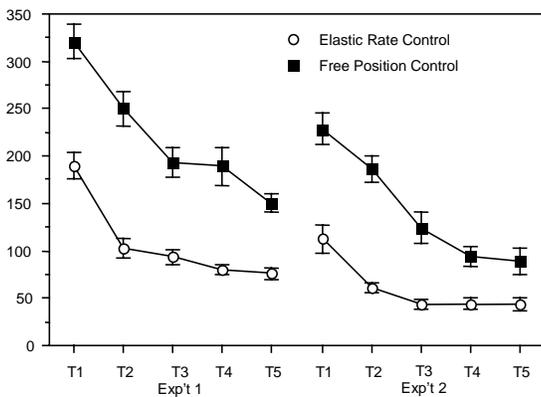


Fig. 7. Mean translation inefficiency (%)

Secondly, the integral transformation in rate control is a low pass filter that removes the higher frequency noise,

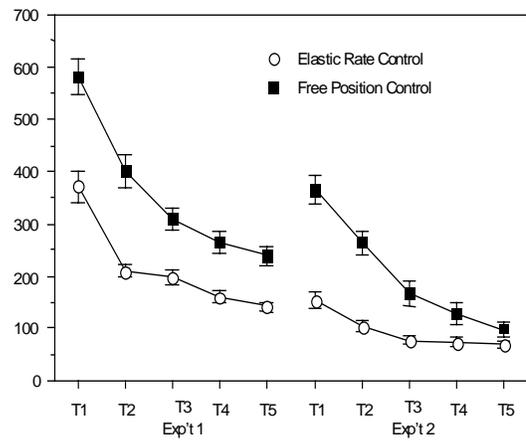


Fig. 8. Mean Rotation Inefficiency (%)

Between translation and rotation. As shown in Fig. 9, the trend in subjects' coordination between total translation and total rotation was similar to that of translation or rotation. The rate control group was significantly more efficient than the position control group in both

experiments, although the magnitude of the difference was reduced after the instruction change during Experiment 2. At the last test, the inefficiency of the rate and position control group was 26.7% and 36.8% respectively. Interestingly, the mean percentages of “wasted” movement in the translation-rotation space were in fact lesser than in the translational space and rotational space, suggesting that there is little reason to separate translation and rotation control into two hands, as in some telerobotic systems.

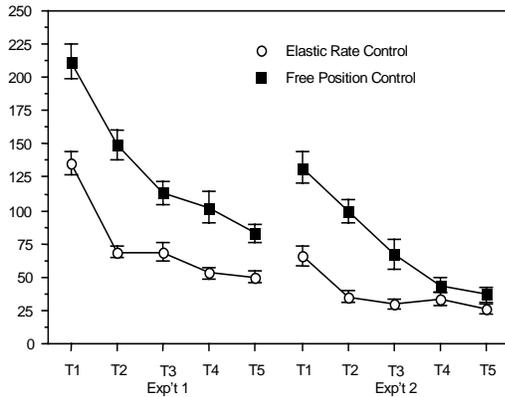


Fig. 9. Inefficiency in translation-rotation space

Total Transport. The above three measures separately indicated users’ efficiency with the two 6 DOF input devices in translation, rotation, and between translation and rotation. The total transport, defined as the line integral of the four vertices of the cursor tetrahedron, was used as an integrated measure of coordination in 6 degrees of freedom. Same as the conclusions drawn from the previous three measures, the rate control device was significantly more efficient (or more coordinated) than the position control device. Practice and instructional emphasis improved efficiency with both devices and the difference between the two was reduced by the instruction, but the final difference was still significant. At the last test of the Experiment 2, the inefficiency of the rate control group was 65.4% and that of the position control group was 96%.

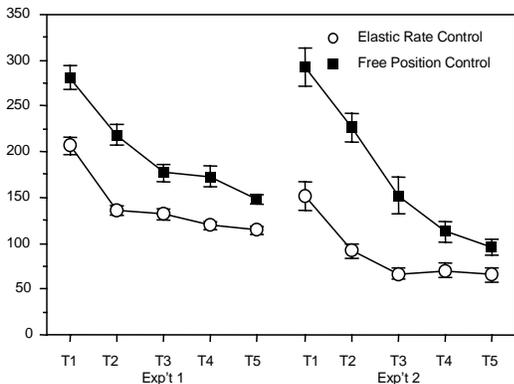


Fig. 10. Inefficiency in total transport

The impact of 3D mental rotation. As recent studies in mental rotation have shown [10], human subjects are incapable of mentally rotating objects in 3D space. This is particularly true when the rotation axis does not coincide with viewer’s primary axes. In Parson’s experiments, subjects did not perform better than chance in mentally finding the correct rotations about arbitrary 3D axes. In our experiments, it is indeed true that the subjects were significantly less efficient in trials with arbitrary initial rotation mismatch than in trials with rotation mismatch about primary axes. Fig. 11 shows such an impact.

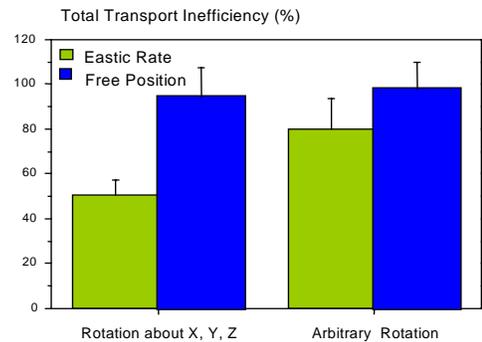


Fig. 11. The impact of rotation type on total movement (total transport data from the last test of Experiment 2)

In not a single trial in our experiments, however, were the subjects unable to successfully complete a trial both in translation and in rotation, including trials in practice sessions when the rotation mismatches are randomly generated and not repeated. The reason, in our view, lies in interaction, the interaction between action and cognition. Note that to be able to manipulate still needs the involvement of mental rotation, the probability to match the target by random manual exploration could not be high. We should point out that the focus of this study is not on mental rotation, although interesting research can be carried in such a direction with the current paradigm.

A greater efficiency difference between the two devices was found when the trials involved rotations only about viewer’s primary axes. This is again plausible: for arbitrary rotation trials, the rotation inefficiency was partially caused by physically searching the correct rotations. For trials that were mismatched about the primary rotation axes, subjects can mentally find the correct rotation more easily so the advantage of the rate control device that may enable higher degree of coordination was better revealed in such trials.

CONCLUSIONS AND GENERAL DISCUSSIONS

The efficiency based coordination measure

Although the experiment showed that the efficiency based measure of coordination was sensitive enough to reveal performance differences, this is not necessary the only “correct” measure to quantify coordination in multiple

degrees of freedom. One possible argument against such a measure is that it is a *definition*, not a validated conclusion about coordination. The critical issues are whether it is an arbitrary definition or a definition in agreement with our common sense judgement of coordination, and furthermore, if the definition is informative. We think both are true. Another drawback of the efficiency measure is that other factors besides manual coordination, such as the mental rotation factor presented above also contribute to the trajectory efficiency, although one can argue that coordination simply includes a cognitive component.

The coordination measure proposed in the paper can be applied to research beyond 6 DOF input devices. For instance, it is conceivable to define coordination of human movement (such as arm movement) that involves N joints. If we defined N dimensional coordination space with each axis as the distance from the angular position of a joint to its goal position, then a perfectly coordinated movement should result in a straight line from the initial mismatch to (0, 0, ...0) in that space. Similarly, we can also define coordination for two-handed computer input.

Isomorphic manipulation vs. tool operation

By applying the efficiency based coordination measure to input device evaluation, we begin to gain insights into the characteristics of 6 DOF input devices that have not been rigorously demonstrated before. Our experiments showed that while the 6 DOF free moving position control device was faster in docking task completion, the elastic rate control device produced more efficient or coordinated trajectories. The difference was true even after emphasis on coordination was explicitly given to the subjects. The contrast between the pros and cons of the two types of devices tested illustrate a more general philosophical issue on computer input device design: isomorphism (direct manipulation) versus tool-using that has been informally debated by researchers [5]. As shown in Fig. 12, there is in fact a continuum on the dimension of directness². The most dominating factor to directness of an input method is the transformation from the *control space* to *display space*. The more mathematically complex this transformation is, the more indirect the input technique is. Input techniques with first order (rate control) or higher order control dynamics are indirect “tools”. With these techniques, one or more integrals are involved in the mathematical mapping from the control space (user’s control actions) to the display space (cursor

movements). The elastic rate controller (EGG) used in the experiments is such an example.

Moving to the left of Fig. 12, input devices become more direct. For position control techniques, the mathematical transformation from the control space to the display space is a multiplication, which is simpler than integration. Among position control techniques, *absolute* devices, such as a 2 DOF digitizing tablet or the 6 DOF Fingerball in Experiment 4 are more direct than *relative* devices, such as a 2 DOF mouse or the 6 DOF glove [16]. Relative devices require a clutch mechanism to engage and disengage the link between control actions and cursor movements. For a mouse, for example, lifting it from mouse pad will disengage the linkage between control and display.

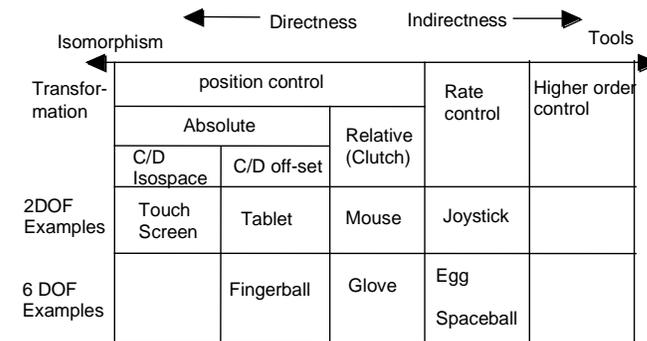


Fig. 12. Isomorphism - tool continuum: A taxonomy of classifying input devices according to directness of transformation from control space to display space

Another factor that affects the directness of position control techniques is the control-display (C-D) ratio. When the C-D ratio is 1, the multiplication operation is reduced to an assignment (copying) operation, which makes the input control more direct than when the C-D ratio is not 1.

There is still another factor that makes some absolute position input techniques more direct than the others: the orientation or location *offset* between the control space and the display space. Both a touch-screen and a tablet are absolute position control devices but the latter has an offset between the display and the control space in orientation (about 90° in pitch) and in location (about 20 - 40 cm in the vertical and/or in the horizontal axes). A touch screen interface is therefore more direct than a tablet interface. In the experiments presented in this paper, all input techniques had a translation offset between the control space and the display space, but no orientation offset. 6DOF techniques without offset can conceivably be implemented, particularly in immersive virtual environments in which the display space (where the user looks) and the control space (where the user moves her limbs) can completely overlap with each other.

It should be noted that to the left of Fig. 12 there are input devices that are even more direct. These are the position control devices with force-reflecting capabilities. The

² Fig. 12 can be viewed as an input device design space or taxonomy. For proposals and discussions of input taxonomy, see W. Buxton “Lexical and Pragmatic Considerations of Input Structure” *Computer Graphics* 17 (1); J. Mackinlay, S.K. Card, G.G. Robertson “A Semantic Analysis of the Design Space of Input Devices” *Human-Computer Interaction* vol 5 pp145-190.

ultimate isomorphic input controller is one that allows force feedback in all directions, to recreate what we would feel when manipulating real 3D objects directly with our bare hands. In other words, the ultimate isomorphic interfaces are completely "transparent" to the user.

It is important to note that there are both advantages and disadvantages to techniques on each end of the isomorphism - tool continuum, as illustrated by our coordination experiments. In daily life, we prefer to perform many tasks with our bare hands. Even with a glove, the small "transformation" between the hand and the actual manipulation may be undesirable on some occasions. On the other hand, we do frequently use various tools, sometimes as simple as rulers, wrenches, screwdrivers, etc., for precision, for power and for overcoming some of our other physical limitations. In general, more isomorphic (more direct) designs are more intuitive and require less learning. Such devices are needed for applications where an explicit learning period is perhaps not available, such as commercial video games where users should be able to walk-up and play immediately. The disadvantages with such isomorphic designs lie in possible fatigue, coarseness of the control action, and anatomical limitations of the human limb. In contrast, less direct, tool-like devices may take more time to learn but may be more efficient in terms of reduced fatigue, coordinated motions, and fewer physical limitations of the human limb. Such designs can be more suitable for tasks of long duration, such as in teleoperation and image visualization.

ACKNOWLEDGEMENTS

The experiments presented here were conducted at the ETC lab of the University of Toronto. The study was made possible by a research grant from ITRC, a center of excellence of Ontario. We thank our co-grantee Bill Buxton for his support and close collaboration. We also like to thank the IBM Almaden Research Center for supporting the first author to continue and complete this study.

REFERENCES

1. Altmann, S. L. (1986). *Rotations, Quaternions, and Double Groups*. Oxford: Clarendon Press.
2. Accot, J. and Zhai, S., Beyond Fitts' Law: Models for trajectory-based HCI tasks, *Proc. CHI'97*, 295-302.
3. Ellson, D. C. (1947). *The independence of tracking in two and three dimensions with the B-29 pedestal sight*. Report TSEAA-694-2G, Aero Medical Laboratory.
4. Flash, T., Hogan, N. The coordination of arm movements: An experimental confirmed mathematical model, *The Journal of Neuroscience*, Vol 5, No. 7. 1688-1703, 1985.
5. Green, M., Bryson, S., Poston, T. & Wexelblat, A, Hands off my VR: the role of gestures in VR (panel session). *Proc. of Virtual Reality Software and Technology (VRST 1994)*, 267-268.
6. Hinckley, K., Tulio, J., Pausch, R., Proffitt, D, Kassell, N., Usability Analysis of 3D Rotation Techniques, to appear *Proc. of ACM Symp. UIST'97*, 1997.
7. Jacobus, H. N., Riggs, A. J., Jacobus, C. J., & Weinstein, Y. (1992). Implementation issues for telerobotic handcontrollers: human-robot ergonomics. In M. Rahimi & W. Karwowski (Eds.), *Human-Robot Interaction*. London: Taylor & Francis.
8. Kabbash, P., Buxton, W., Sellen, A., Two-handed input in a compound task, *Proc. CHI'94*, 417-423.
9. O'Hara, J. (1987). Telerobotic control of a dexterous manipulator using master and six-DOF hand controllers for space assembly and servicing tasks. *Proc. Human Factors Society 31st Annual Meeting*.
10. Parsons, L., Inability to reason about an object's orientation using an axis and angle of rotation. *Journal of Experimental Psychology: Human Perception and Performance*, 1995, Vol.21, No.6, 1259-1277.
11. Poulton, E.C., Unwanted asymmetrical skill transfer effects with balanced experimental designs. *Psychological Bulletin*, 66 1 (1966), 1-8.
12. Rice, J. R., Yorchak, J. P., & Hartley, C. S. (1986). *Capture of satellites having rotational motion*. *Proc. Human Factors Society 30th Annual Meeting*
13. Senders, J. W., Christensen, J. M., & Sabeh, R. (1955). *Comparison of single operator's performance with team performance in a tracking task* (TN-55-362): Aero Medical Laboratory, Wright Air Development Center.
14. Ware, C. Using hand position for virtual object placement, *the Visual Computer*, 1990, Vol 6, 245-253
15. Zhai, S. (1995). *Human Performance in Six Degree of Freedom Input Control*. Ph.D. Thesis, Univ. of Toronto.
[Http://vered.rose.toronto.edu/people/shumin.html](http://vered.rose.toronto.edu/people/shumin.html)
16. Zhai, S., Milgram, P. Buxton, W., The influence of muscle groups on performance of multiple degree of freedom input control, *Proc. of CHI'96*.
17. Zhai, S., Senders, J.W., Investigating Coordination in Multidegree of Freedom Control I: Time-on-Target Analysis of 6 DOF Tracking, to appear in *Proc. Human Factor Ergonomics Society 1997*.
18. Zhai, S., Senders, J.W., Investigating Coordination in Multidegree of Freedom Control II: Correlation Analysis of 6 DOF Tracking. *ibid*

Hands-Free Multi-Scale Navigation in Virtual Environments

Joseph J. LaViola Jr.

Daniel Acevedo Feliz

Daniel F. Keefe

Robert C. Zeleznik

Brown University
Department of Computer Science, Box 1910
Providence, RI 02912
{jjl, daf, dfk, bcz}@cs.brown.edu

Abstract

This paper presents a set of interaction techniques for hands-free multi-scale navigation through virtual environments. We believe that hands-free navigation, unlike the majority of navigation techniques based on hand motions, has the greatest potential for maximizing the interactivity of virtual environments since navigation modes are offloaded from modal hand gestures to more direct motions of the feet and torso. Not only are the users' hands freed to perform tasks such as modeling, notetaking and object manipulation, but we also believe that foot and torso movements may inherently be more natural for some navigation tasks. The particular interactions that we developed include a leaning technique for moving small and medium distances, a foot-gesture controlled Step WIM that acts as a floor map for moving larger distances, and a viewing technique that enables a user to view a full 360 degrees in only a three-walled semi-immersive environment by subtly amplifying the mapping between their torso rotation and the virtual world. We formatively designed and evaluated our techniques in existing projects related to archaeological reconstructions, free-form modeling, and interior design. In each case, our informal observations have indicated that motions such as walking and leaning are both appropriate for navigation and are effective in cognitively simplifying complex virtual environment interactions since functionality is more evenly distributed across the body.

CR Categories and Subject Descriptors: I.3.6 [Computer Graphics]: Methodology and Techniques - Interaction Techniques; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism - Virtual Reality

Additional Key Words: Navigation techniques, Auto Rotation, Gestural Interaction, Virtual Reality

1 Introduction

Each year computational power inexorably increases and virtual environments become richer as increasingly realistic displays of virtual scenes become viable. Despite this continuing progress in visual realism, the ability of a user to engage in rich interactions in a virtual environment fails to follow such a steady advance. In fact, it can be argued that the quality of user interaction moves in opposition to the complexity of virtual environments because the number



Figure 1 A user examining the Step WIM.

of possible tasks increases whereas the fundamental capacities of the human remains constant. To address the issue of increasing task complexity, we must consider techniques that make better use of the finite set of human capabilities. The common approaches to increasing task range and complexity are to either squeeze more data out of an existing human channel, perhaps by distinguishing more or different gestures, or to offload task complexity from one overloaded channel to a different channel. In the context of virtual environment interaction, the former approach has been extensively applied to hand gestures as exemplified by the interaction techniques of SmartScene[20]. The latter approach has received increased attention and has achieved notably positive results, for example, when task details that are tedious to express with gestures alone are naturally given through speech commands[3].

The goal of this paper is to present techniques for offloading a range of virtual environment navigation techniques onto previously under-exploited human capabilities for walking, leaning, bending and turning. Certainly most virtual environments allow users to do all the above actions with effects similar or identical to those if performed in the physical world. Our techniques, however, extend and amplify the effects of those actions additionally to support multi-scale navigation through virtual environments and full 360 surround viewing of semi-immersive environments consisting of only three vertical walls.

1.1 Organization

The remainder of this paper is organized in the following manner. First, we discuss previous work related to navigation in virtual environments. Second, we describe the Step WIM (see Figure 1), a tool for quickly traveling to any part of a virtual world, and describe the interaction techniques used to invoke, scale and dismiss it. Third, we describe our leaning technique for navigating short to medium range distances. Fourth, we present auto rotation, a technique for

automatically viewing a full 360 degrees given the constraints of a three-walled display. Finally, we discuss future work and our conclusions.

2 Previous Work

Previous techniques for navigation within virtual environments have covered a broad categorization of approaches ranging from directly manipulating the environment with hand gestures, to indirectly navigating using hand-held widgets, to simulating physical metaphors, to identifying body gestures, and even to recognizing speech commands. Although no systematic study has attempted to evaluate this gamut of controls, Bowman presented preliminary work[1] that evaluated a smaller class of immersive travel techniques and discussed relevant considerations for the design of new techniques. Our review of techniques focuses on general-purpose unconstrained, floor-based navigation controls, although we note the relevance of some application-specific, constrained navigation controls, such as Galyean's guided navigation technique[10].

Perhaps the most prevalent style of navigation control for virtual environments is to directly manipulate the environment with hand gestures. For example, SmartScene provides clever controls for the user to navigate through a virtual world by treating the world as one large object that can be gesturally grabbed, moved and scaled with both hands to achieve the effect of user navigation[20]. Others demonstrated techniques for orbiting about and moving relative to objects specified by hand gestures[15][18]. All of these techniques can be quite effective yet, in complex applications with large sets of interaction tasks, hand-worn or hand-held devices are often overloaded making them more difficult to learn and use.

Widget-based controls are also popular for navigating within virtual environments[7][9][23]. The most relevant widget to our work is Stoakley's implementation of a navigation technique based on "flying" into a hand-held world-in-miniature (WIM)[21]. This technique allows a user first to indicate a desired new viewing location using a hand-held miniature representation of the virtual environment, and second to be seamlessly "flown" to that location by an animated transformation of the hand-held WIM. The primary disadvantage of this and most other widget-based techniques used for navigation is that they again require the involvement of the users hands. Our Step WIM technique provides two important benefits over Stoakley's hand-held WIM. First, it offloads all WIM interactions onto the user's feet and body thereby freeing the hands to perform other interaction tasks. Second, the Step WIM, in addition to working in head mounted displays, also is effective in Cave-based projective environments because the full resolution of the floor is available and the user's hands do not obstruct the WIM display.

A less-generally applied alternative to hand-controlled techniques, is to control navigation with body gestures often coupled to mechanical devices. Darken explored the use of an omni-directional treadmill[4] that with only limited success enabled a user to navigate by walking. Brooks[2] and Iwata[11] have also developed treadmill-based navigation techniques. Others have provided unicycles, bicycles or automobile shells that allow at least some aspects of navigation, such as speed or bearing to be controlled with the feet or a body posture. These techniques tend to be quite restrictive and are generally appropriate for only specific types of simulation environments such as military battlefield training.

A more general although less frequently explored approach is to map body gestures directly to virtual environment navigation. Fuhrmann, et al[8] developed a head-directed navigation technique in which the orientation of the users head determined the direction and speed of navigation. Their technique has the advantage of requiring no additional hardware besides a head tracker, but has



Figure 2 The Step WIM widget which allows users to quickly navigate anywhere in the virtual world. The small sphere by the user's foot indicates his position in the miniature.

the disadvantage that casual head motions when viewing a scene can be misinterpreted as navigation commands. In addition, a severe drawback of this and other head-based techniques, as Bowman discusses[1], is that it is impossible to perform the common and desirable real-world operation of moving in one direction while looking in another. An alternative technique that is often based on head-tracking[22] is to control navigation by walking in place[19]. The speed of movement is coupled to the user's pace, but again the direction of motion is restricted to the user's head orientation. The Placeholder VR system[12] allowed a user to walk in place but determined the direction of motion from the orientation of the user's torso thus allowing the decoupling of the user's head orientation from their direction of movement. We feel that this technique is close in spirit with our hands-free navigation philosophy and perhaps should be integrated with our suite of controls to afford another technique for navigating short distances. Another direct body-based navigation technique is found in the Osmose system which uses sensors to measure the tilt of the user's spine and the expansion of their chests[5]. These measurements allow users to move horizontally with body tilt and up and down with chest expansion and contraction.

The final category of techniques for motion control is based on speech recognition. Speech allows a user to modelessly indicate parameters of navigation and can often be used in conjunction with gestures to provide rich, natural immersive navigation controls[3]. We believe that speech controls should play a role in virtual environment navigation, but we also feel that it is also critical to support effective, speech-free navigation techniques for the common situations where speech recognition is unavailable, inappropriate or simply not desired.

3 The Step WIM

The Step WIM is a miniature version of the world that is placed on the ground, under the user's feet in the virtual environment. The idea is derived from Stoakley's hand-held World In Miniature which was used for selecting and manipulating virtual objects[21] as well as navigation and locomotion[16]. However, instead of treating the WIM as a hand-held object, we wanted to achieve an effect similar to walking through a miniature environment landscape, such as Madurodam in The Hague. Consequently, when a user invokes the Step WIM, a miniature version of the virtual environment is placed beneath their feet such that the actual position of the user in the virtual environment coincides with the approxi-

mate location of the user's feet in the miniature (see Figure 2). The Step WIM then functions as an augmented road map. The user can either walk around the Step WIM to gain a better understanding of the virtual environment, or he or she can use the Step WIM to navigate to a specific place by simply walking to a desired location in the WIM and invoking a scaling command, causing the Step WIM to animate scaling up around the user's feet¹, thereby seamlessly transporting the user to the specified virtual environment location. As Bowman[1] and Pausch[16] discuss, animation of the Step WIM is essential to the user's sense of location. In situations where the Step WIM is either too large or too small, the user can, upon command, increase or decrease the size of the wim.

3.1 Invoking, Scaling and Dismissing the Step WIM

In addition to the effect of walking through a virtual environment in miniature, a second critical aspect of the Step WIM is that it can be entirely controlled with a single foot gesture. We determined that a single gesture is sufficient for controlling all three operations of invoking, scaling, and dismissing the Step WIM by performing an informal Wizard of Oz experiment. In this experiment, we asked six people to control the Step WIM by tapping their foot and saying what they wanted the Step WIM to do. When the Step WIM wasn't displayed, it was clear that tapping was meant to invoke the Step WIM. When the Step WIM was displayed, we observed that users looked down at the Step WIM when they wanted tapping to transport them to a new location, but looked away from the Step WIM when they wanted tapping to dismiss the Step WIM.

Based on this very informal experience, we prototyped a number of gestures for controlling the Step WIM. From this set of prototypes, we culled out two different styles of foot gestures that we feel are both natural and robustly recognizable. We outline each of the two gestures both because each gesture requires different sensing technology and because we have not yet conducted a formal evaluation to clarify the advantages of one gesture over the other. In either case, we disambiguate the user's intention to either dismiss the Step WIM or transport him or herself to a new location by estimating the gaze direction when the foot gesture is performed. If the user's head direction is 25 degrees below horizontal (i.e., he or she is looking down at the Step WIM), then we determine the user is trying to move to a new location, otherwise we determine that the user wants to dismiss the Step WIM.

3.1.1 The Foot-Based Interface

In our Cave, a four-sided (three walls and a floor) semi-immersive projection-based virtual environment, users have to wear slippers over their shoes to protect the display floor from scuff marks and dirt from the outside world. Therefore, we developed a pair of Interaction Slippers, shown in Figure 3, that are instrumented to make it easy to identify toe and heel tapping. We were also inspired by the scene in *The Wizard of Oz* where Dorothy taps her heels to return to Kansas.

To invoke the display of the Step WIM with these Slippers, the user taps his or her toes together, establishing a conductive cloth contact which is easily sensed and treated as a "button" press. Once displayed, the user can move to a new location by simply walking to a desired place in the Step WIM and clicking the toes together again, while looking at the Step WIM. To dismiss the Step WIM,

¹The world actually scales up around the projection of the user's head onto the floor, as indicated by a small green icon. By tracking the head instead of the feet, we avoid obscuration issues with a projected display and afford the user fine-grained control of his or her location in the Step WIM via head movement.



Figure 3 The Interaction Slippers allow users to tap either their toes or heels to trigger Step WIM functionality.

the user makes the same clicking gesture while looking away from the floor.

Two important design considerations when creating the Interaction Slippers were that they be both comfortable and untethered. We addressed these considerations by embedding a Logitech Trackman Live!TM wireless trackball device that uses digital radio technology[13] into a pair of commercially available slippers. We chose wireless radio technology over other approaches, such as infrared, because it provides a range of up to 30 feet, and does not require unoccluded line-of-sight to a sensor. We inserted the Trackman into a hand-made pouch on the right slipper and rewired two² of the Trackman's three buttons by connecting each one to a pair of conductive cloth[14] patches on the instep of the right slipper. On the instep of the left slipper, we placed two more conductive cloth patches. Touching a cloth patch on the left slipper to a cloth patch pair on the right slipper completes the button press circuit. This design enables us to distinguish two gestures corresponding to heel and toe contacts respectively.

3.1.2 Body Controlled Gestural Interface

The alternative interface that we present for controlling the Step WIM works exactly the same as the toe-based controls except the gesture is an upward bounce instead of a toe tap. An upward bounce gesture is detected whenever the user rises on the balls of his or her feet and then quickly drops back down again. Although this gesture can, in theory, be detected solely through head tracking, we employ a simpler gesture recognition algorithm that uses a waist tracker. Waist tracking is accomplished by having the user wear a conventional belt that has a magnetic tracker mounted on the belt buckle. The advantage of a waist tracking gesture recognizer is that it will not inadvertently classify a bouncing head motion (e.g., looking down and up) as a bounce gesture in the same way a head-based gesture recognizer might. There is a clear disadvantage to wearing an augmented belt to track waist position since the user must wear another tethered device, but waist tracking is required for other parts of our interface (See Section 4), so we simply take advantage of the availability of this more robust data.

Our algorithm for recognizing a bounce gesture is initially calibrated by storing the user's waist height h from the tracker attached to the user's belt. We record this value in the display device's coordinate system, since that frame of reference will remain constant as the user moves through the virtual environment. We then monitor each tracker data record, checking whether the user's waist is above the initial waist calibration height by more than a distance of Δh . We found a Δh of 1.5 inches to work well. We then accumulate the amount of time, t_{up} , in which the waist height is above the given

²Our current implementation of interaction slippers utilizes only two of three Trackman buttons. In future work we plan to use of the third button as well as the trackball.

$h + \Delta h$. If t_{up} is between a threshold, t_{min} and t_{max} we consider this a bouncing gesture.

3.2 Step WIM Scaling

For many environments, a single-sized Step WIM is sufficient to encompass the entire virtual environment. However, some environments are so large that a single-sized Step WIM is inadequate for both providing access to the entire virtual environment and for providing enough detail to accurately control navigation. For example, if the Step WIM for a large environment is scaled to provide reasonable detail, then it will not fit within the physical walking area of the user (in the case of our Cave, an 8 foot square). Alternatively, if the Step WIM is scaled to fit within the user’s physical walking area, there may not be enough detail for the user to control navigation precisely.

We present two additional Step WIM controls that address this problem. The first control allows the user to interactively change the scale of the Step WIM. The second control, presented in Section 4, allows the user to navigate the Step WIM, thus providing the user with access to distant regions of the Step WIM that were not previously within his or her physical walking area.

Although the control for Step WIM scaling requires just a single additional gesture, we again provide two different gestures corresponding to whether or not the Interaction Slippers are used.

3.2.1 Foot-Based Scaling

When wearing Interaction Slippers to control the Step WIM, the user activates and deactivates Step WIM scaling mode by clicking the heels together (as distinguished from clicking the toes together). This scaling mode overrides the previous Step WIM controls until it is deactivated by a second heel click. When the user first enters Step WIM scale mode by making heel cloth contacts, the user’s head position is projected onto the Step WIM and stored. This projected point is used as the center of scale for changing the size of the Step WIM. As the user walks, instead of moving about within the Step WIM, the Step WIM is translated, so that the center of scale always lies at the projection of the user’s head onto the floor (See Figure 4). In addition, if the user walks forward with respect to the Step WIM, as if to get a closer look, the Step WIM gets larger. If the user walks backward within the Step WIM, as if to see a larger picture, the Step WIM scales smaller. To return to the standard mode of controlling the Step WIM with toe taps, the user must again click his or her heels together. This second heel click freezes the scale of the Step WIM until the next time the user enters Step WIM scale mode.



Figure 4 A user prepares to scale the Step WIM upward using the Interaction Slippers (left). As the user moves forward the Step WIM gets larger and his position in the miniature is maintained (right).

3.2.2 Body Controlled Gestural Scaling

Alternatively, when controlling the Step WIM with bounce gestures, the user can change the scale of the Step WIM directly with-

out having to enter a special scaling mode. The control for scaling the Step WIM smaller about the projection of the user’s waist is for the user to simply rise up on the balls of his or her feet for longer than the bounce time threshold, t_{max} . Once this time threshold has been exceeded, the Step WIM will start to shrink at a rate of 2 percent per second. To make the Step WIM larger, the user assumes a slight crouching posture by bending his or her knees enough to lower the waist position by Δh . Once again, if this posture is held for longer than t_{max} , the Step WIM will begin to grow at a rate of 2 percent per second.

As an example of when changing the size of the Step WIM is useful, consider a large virtual environment the size of the Earth. The user can scale the Step WIM down so that the entire map of the Earth fits within the physical space of the available walking area. Then the user can walk roughly to a position of interest, perhaps a country, and then partially scale the Step WIM up about that position in order to more easily identify a desired location, perhaps a city, before finally transporting him or herself to that location by scaling the Step WIM up completely.

4 Navigation By Leaning

The Step WIM controls provide a hands-free navigation technique for moving medium to large distances through virtual environments. To also support hands-free navigation of small to medium distances, we refined the navigation-by-leaning technique proposed by [6] to reduce the likelihood of accidentally recognizing a relaxed posture as a leaning action. Our leaning technique allows the user to navigate through the virtual environment by simply leaning at the waist in the desired direction of movement. An added benefit of leaning controls over head-based navigation techniques is that the user can look in a direction that is different than the one in which he or she is moving. For example, in an art gallery environment the user can lean in order to move along a wall of pictures while always concentrating on the picture immediately in front of him or her (See Figure 5).

In addition to navigating relatively small distances by leaning, the user can also lean to translate the Step WIM to gain access to Step WIM locations that would not otherwise fit within the user’s physical walking area. The decision to move the Step WIM widget by leaning instead of navigating through the virtual environment depends on whether the Step WIM is active and whether the user’s gaze direction is 25 degrees below the horizontal (i.e., at the Step WIM).



Figure 5 A user leans to the right (left) to view a painting (right).

To detect the direction and magnitude of leaning, both the user’s waist and head are tracked. We compute the direction of leaning, the *leaning* vector \vec{L}_R , by projecting the vector from the waist to the head onto the horizontal floor plane. To map \vec{L}_R to navigation controls, we simply map the navigation heading to the direction of \vec{L}_R and we map the navigation speed to $\|\vec{L}_R\|$, such that the more the user leans in a given direction, the faster he or she will go.

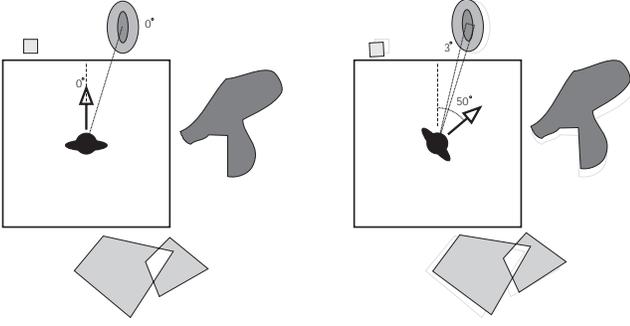


Figure 6 An illustration of the auto rotation technique. As the user rotates to the right, the world auto rotates in the opposite direction based on the scaled 2D Gaussian function (see equation 5).

The actual mapping function we use between $\|\vec{L}_R\|$ and navigation speed is given by a function that is dependent on where the user is located with respect to his or her walking area. This position dependence derives from observations of how people work in a virtual environment with a relatively small walking area. Typically, we find that people choose to walk in the direction they want to go until they cannot walk any further, at which point they switch to a navigation technique. Therefore, our mapping function is most sensitive to leaning in a given direction when the user cannot physically move any farther in that direction. This varied sensitivity to leaning makes navigation control more robust since accidental leaning caused by normal human head and body postures tend to be de-emphasized. For example, when users want to move forward, they will have to lean farther forward, if they are standing in the center of their physical walking area, than if they are already close to the front of it. Thus we fine-tuned our leaning function so that inadvertent variation in the user's leaning is essentially discarded when the user is at the center of the working area, while the user needs to lean only subtly in the direction he or she wants to go when already close to a boundary (i.e., a wall of the Cave).

Our position dependent function is a linear function which provides the minimum amount L_T the user has to lean to produce a translation.

$$L_T = a \cdot D_{min} + b \quad (1)$$

where D_{min} is the minimum distance between the user and a physical boundary in the direction he or she is leaning.

We calculate a and b given the minimum leaning value L_{min} when the distance to a boundary is close to 0, and the maximum L_{max} , when the distance is equal to the diagonal of our Cave, approximately 11.3'. Table 1 shows the values we have used for these thresholds, L_{min} and L_{max} , both when leaning forward and backwards³ and the corresponding *slope* and *y-intercept* values, a and b , used in each case.

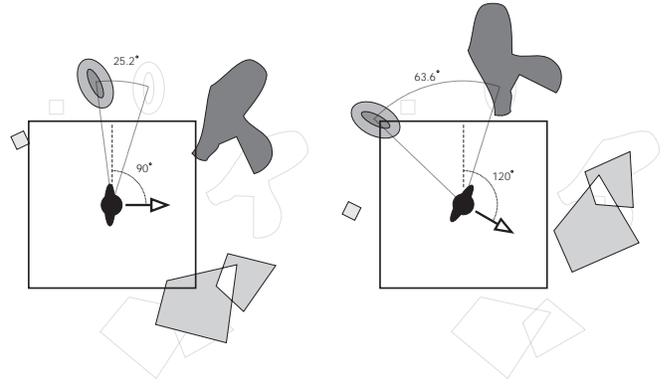
The user's unscaled velocity v is then calculated by

$$v = \|\vec{L}_R\| - L_T. \quad (2)$$

4.1 Exponential Mapping For Navigation Velocity

During informal user testing of the leaning control, we noticed that when users wanted to move somewhere in the virtual world their

³We decided to modify the thresholds when the user is leaning backwards, since it is a more difficult gesture to perform.



	Leaning Forward	Leaning Backwards
L_{min}	4.5''	1.0''
L_{max}	7.0''	7.0''
a (Slope)	0.018	0.044
b (Y-Intercept in feet)	0.375	0.083

Table 1 The minimum and maximum leaning thresholds used to calculate the slope and y-intercept to our linear mapping function.

gaze was generally focused on the place they wanted to go even as this location was moving towards them. Since objects in our virtual environments are generally lower than the user's head height, we improved our mapping function by recognizing that, as distant objects come closer, the user's head naturally tilts down to maintain focus. Thus, we correlate the rate of movement to the user's head orientation with respect to the vertical axis such that the movement rate is exponentially decreased as the user's head rotates increasingly downward even though the amount of lean is constant. This exponential mapping has proven useful, especially for navigating the Step WIM, since the object of focus appears to smoothly decelerate as it draws near and the user's head tilts further down to see it. In general, the user's head orientation can be thought of as a vernier control that modifies the rate of movement indicated by the user's amount of lean.

We have found that a scaled exponential function

$$F = \alpha e^{-\beta |\vec{head} \cdot \vec{V}_{up}|} \quad (3)$$

where α is the maximum speed factor, β defines the steepness of the exponential curve, \vec{head} is the user's head orientation vector, and \vec{V}_{up} is the vertical vector coming out of the display floor, provides smooth translations and works well in the different environments we have tested.

The final leaning velocity is calculated by

$$v_{final} = F \cdot v \quad (4)$$

which is then applied to the leaning vector L_R . The coefficients for the exponential function, α and β , change depending upon the scale at which the navigation is taking place. When users are leaning to navigate the virtual world, values of 3.0 and 6.0 for α and β provide a good fall-off for movement as they focus on a point closer to their position in our virtual environment. For translating the Step WIM, these values were different since the user is mostly looking down towards the floor. When the size of the Step WIM exceeded the

physical space of our Cave, values for α equal to 2.5 and β equal to 5.0 worked well.

5 Auto Rotation

In fully immersive virtual environments, there is generally no need to provide any explicit control for rotating the virtual environment relative to the user, since the user can turn to face any direction in the virtual environment. However, a common semi-immersive display configuration is a three-walled Cave which affords only a 270 degree view of the world when the user stands at the Cave center. In such semi-immersive environments, the user generally needs an explicit user interface control for rotating the virtual world so that the full 360 degrees of the virtual environment can be viewed.

To provide a complete suite of hands-free navigation controls for semi-immersive virtual environments with only three walls, we developed an amplified rotation technique, a non-isomorphic 3D rotational technique that implicitly allows a user to view a full 360 degrees even though the physical display environment does not surround the user (see Figure 6). Poupyrev most recently studied non-isomorphic 3D rotational techniques and found them to be effective for hand-held object manipulation[17]. However, applying non-isomorphic rotation to navigation is more complicated, since rapid, unexpected rotation of a virtual environment can easily cause cybersickness. Therefore, we prototyped a number of different techniques that all attempt to make the automatically generated amplified rotation as subtle and gentle as possible.

Our first prototype amplified a user's head rotation by linearly counter-rotating the world such that a 120 degree rotation by the user effectively amounts to a 180 degree rotation of the user in the virtual world. Although this technique allows the user to see a full 360 degrees in a three walled Cave, most of our trial users felt at least some degree of cybersickness after only a few minutes. As our second prototype, we keyed the linear counter-rotation of the world to the orientation of the user's torso instead of his or her head, thus eliminating most of the unnatural world rotation when the user was just glancing about. Despite reduction in cybersickness, this technique felt unnatural to our trial users who generally thought that there was continually too much rotational motion, especially as they walked around the environment.

The final technique that balances most user concerns is to apply a non-linear mapping function in which amplified rotation effectively kicks in only after the user has rotated beyond a threshold that is dependent on the user's orientation vector and position in the Cave.

In order to calculate the correct viewing angle, we first define the user's waist orientation as the angle θ between the waist direction vector and the horizontal vector to the front Cave wall projected onto the floor plane⁴. Next we define d as the distance the user is from the back of the Cave. Using these two variables, we calculate the rotation factor ϕ using a scaled 2D Gaussian function

$$\phi = f(\theta, d) = \frac{1}{\sqrt{2\pi}\sigma_1} \cdot e^{-\frac{(|\theta| - \pi(1-d/L))^2}{2\sigma_2^2}} \quad (5)$$

where σ_1 is a Gaussian height parameter, σ_2 is a Gaussian steepness parameter, L is a normalization constant which is used to lessen the effect of d , and the function's μ value is set to π . Using ϕ , we find the new viewing angle by

$$\theta_{new} = \theta(1 - \phi). \quad (6)$$

⁴We do not take into account whether the user is looking up or down.

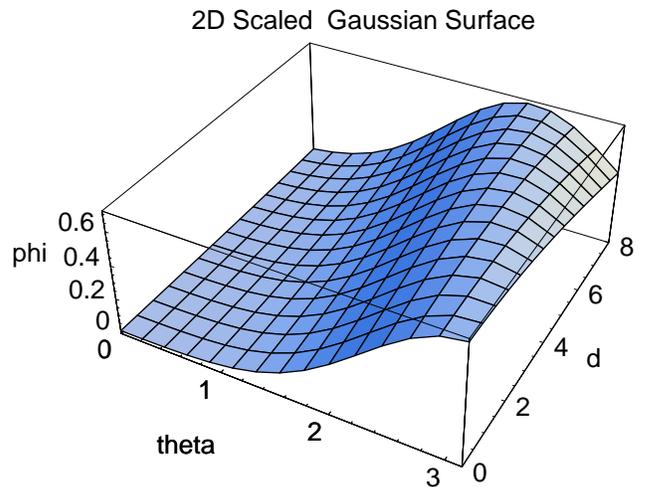


Figure 7 A visual representation of the scaled 2D Gaussian surface we use to find the rotation factor, which determines the degree of rotation amplification. For this graph, σ_1 equals 0.57, σ_2 equals 0.85, and L equals 30.

In order to get a better understanding of what the 2D Gaussian surface does, consider Figure 7. In the figure, we see that if the user is close to the front wall of the Cave, d equal to 0, the user has more visual display in which to rotate about and, as a result, the Gaussian bump is shifted closer to π on the θ -axis, reducing the amount of rotation amplification as the user's rotation angle gets larger. Conversely, if the user is closer to the back of the Cave (d equal to 8), he or she only has 180 degrees of rotation available before looking out of the display. Therefore, we need greater rotation amplification for the user to see a full 360 degrees. So the Gaussian bump is shifted closer to 0 on the θ -axis.

In combination with the leaning metaphor described in the previous section, users can also travel in directions that were originally directly behind them when they faced the front wall of our three-sided Cave by first turning to face either the right or left wall. We have observed that users need time to adjust to this distorted spatial mapping, but can at least navigate in any direction after only a few minutes. However, we have not yet attempted to quantify the effect of this auto rotation technique on a user's sense of spatial relations and this remains an important area of new research.

6 Future Work

We hope to explore a number of specific avenues in future work including improvements to our techniques for tracking the user and extensions to our current suite of interaction techniques.

Our current implementations require that users minimally wear a head and belt tracker; although we believe that it may be possible to robustly perform all operations, except the toe-tapping gestures, with only one accurate head-tracking device. A further improvement would be to completely untether the user by developing appropriate vision or wireless tracking techniques. Furthermore, we believe that our leaning gestures could be made even more subtle by incorporating multiple pressure sensors onto the soles of our Interaction Slippers.

We believe that our current set of controls are adequate for navigating through a broad range of virtual environments, although we believe that additional hands-free controls would be helpful for navigating around specific objects and for navigating through spaces

that do not have a floor-plane constraint. In addition we would like to extend our navigation controls to include the walking in place technique and we would like to explore multi-modal navigation techniques based on speech recognition.

7 Conclusion

We have presented a cohesive suite of hands-free controls for multi-scale navigation through a broad class of floor-constrained virtual environments. Since all our controls are hands-free, virtual environment designers have greater flexibility when mapping additional functionality since the user's hands are completely offloaded.

Specifically, our controls allow a user to move small and medium distances, users can simply lean in the direction they want to move independent of their head orientation. Unlike previous leaning techniques[5][6], our leaning control is modified for robustness to consider both the users' location relative to their physical walking area and their head orientation. To move coarse distances, the user can gesturally invoke an adaptation of a conventional WIM, a *Step WIM*, that is displayed on the floor of the user's physical walking area. The Step WIM is controlled either by toe-tapping with wireless *Interaction Slippers* or by tip-toe bouncing gestures. The Step WIM is further extended to support interactive scaling using heel-tapping or crouching gestures, and to support translational movement by leaning.

Acknowledgments

Special thanks to John Hughes, Anne Spalter, Tomer Moscovich, and Andries van Dam. This work is supported in part by the NSF Graphics and Visualization Center, IBM, Advanced Network and Services, Alias/Wavefront, Microsoft, Sun Microsystems and TACO.

References

- [1] Bowman, D. A., Koller, D., and Hodges, L. F. Travel in Immersive Virtual Environments: an Evaluation of Viewpoint Motion Control Techniques. In *Proceedings of IEEE VRAIS'97*, 45-52, 1997.
- [2] Brooks, F. P. Walkthrough: A Dynamic Graphics System For Simulating Virtual Building. In *Proceedings of the 1986 Workshop in Interactive 3D Graphics*, 9-21, 1986.
- [3] Cohen, P. R., Johnston, M., McGee, D., Oviatt, S., Pittman, J., Smith, I., Chen, L., and Clow, J. (1997). QuickSet: Multi-modal interaction for distributed applications, In *Proceedings of the Fifth International Multimedia Conference (Multimedia '97)*, ACM Press, 31-40, 1997.
- [4] Darken, R., Cockayne, W. and Carmein, D. The Omni-Directional Treadmill: A Locomotion Device for Virtual Worlds. Interaction. In *Proceedings of UIST'97*, ACM Press, 213-222, 1997.
- [5] Davies, C. and Harrison, J. Osmose: Towards Broadening the Aesthetics of Virtual Reality. *Computer Graphics* 30(4): 25-28, 1996.
- [6] Fairchild, K., Hai, L., Loo, J., Hern, N. and Serra, L. The Heaven and Earth Virtual Reality: Design Applications for Novice Users. In *Proceedings of IEEE Symposium on Research Frontiers in Virtual Reality*, 47-53, 1993.
- [7] Fisher, S. The AMES Virtual Environment Workstation (VIEW). In *SIGGRAPH 89 Course Notes #29*, 1989.
- [8] Fuhrmann, A., Schmalstieg, D. and Gervautz M. Strolling through Cyberspace with Your Hands in Your Pockets: Head Directed Navigation in Virtual Environments, In *Virtual Environments '98 (Proceedings of the 4th EUROGRAPHICS Workshop on Virtual Environments)*, Springer-Verlag, 216-227, 1998.
- [9] Furness, T. Configuring Virtual Space for the Super Cockpit. *Human Interface Technology (HIT) Laboratory Technical Report*, University of Washington, HITL-M-89-1, 1989.
- [10] Galyean, T. Guided Navigation of Virtual Environments. In *Proceedings of Symposium on Interactive 3D Graphics*, ACM Press, 103-104, 1995.
- [11] Iwata, H. Walking about Virtual Environments on an Infinite Floor, In *Proceedings of Virtual Reality '99*, 286-293, 1999.
- [12] Laurel, B., Stickland, R., and Tow, R. Placeholder: Landscape and Narrative in Virtual Environments, *ACM Computer Graphics Quarterly*, Volume 28, Number 2, May 1994.
- [13] Logitech, <http://www.logitech.com>, 2000.
- [14] Mann, S. Smart Clothing: The Wearable Computer and WearCam. *Personal Technologies*, Volume 1, Issue 1, March, 1997.
- [15] Pierce, J., Forsberg, A., Conway, M., Hong, S., Zeleznik, R. and Mine, M., Image Plane Interaction Techniques in 3D Immersive Environments. In *Proceedings of Symposium on Interactive 3D Graphics*, ACM Press, 39-43, 1997.
- [16] Pausch, R., Burnette, T., Brockway, D. and Weiblen, M. Navigation and Locomotion in Virtual Worlds Via Flight into Hand-Held Miniatures. In *Proceedings of SIGGRAPH 95*, ACM Press, 399-400, 1995.
- [17] Poupyrev, I., Weghorst, S., and Fels, S. Non-Isomorphic 3D Rotational Techniques. In *Proceedings of the Conference on Human Factors in Computing Systems, CHI 2000*, 540-547, 2000.
- [18] Mine, M. Moving Objects In Space: Exploiting Proprioception In Virtual Environment Interaction. In *Proceedings of SIGGRAPH 97*, ACM Press, 19-26 , 1997.
- [19] Slater, M., Steed, A., and Usoh, M. The Virtual Treadmill: A Naturalistic Metaphor for Navigation in Immersive Environments. *First Eurographics Workshop on Virtual Reality*, 71-86, 1993.
- [20] SmartScene™ is a product of Multigen, Inc. More information on SmartScene™ is available from Multigen's website at <http://www.multigen.com/products/smarts.com>, 2000.
- [21] Stoakley, R., Conway, M. J., and Pausch, R. Virtual Reality on a WIM: Interactive Worlds in Miniature, In *Proceedings of Human Factors and Computing Systems, CHI'95*, 265-272, 1995.
- [22] Usoh, M., Arthur, K., Whitton M., Bastos R., Steed, A., Slater, M., and Brooks, F. Walking > Walking-in-Place > Flying, in Virtual Environments", In the *Proceedings of SIGGRAPH 99*, ACM Press, 359-364, 1999.
- [23] Ware, C. and Osborne, S. Exploration and Virtual Camera Control in Virtual Three Dimensional Environments. *Computer Graphics* 24(2): 175-183, 1990.

Non-Isomorphic 3D Rotational Techniques

Ivan Poupyrev¹, Suzanne Weghorst², Sidney Fels³

¹ ATR MIC Research Laboratories ² HIT Lab, University of Washington
2-2 Hikaridai, Seika, Soraku-gun
Kyoto 619-02, Japan
(774) 95-1432
poup@mic.atr.co.jp

Box 352142
Seattle, WA 98195, USA
(206) 685-3215
weghorst@hitl.washington.edu

³ Department of ECE,
University of British Columbia
Vancouver, BC Canada, V6T 1Z4
(604) 822-5338
ssfels@ece.ubc.ca

ABSTRACT

This paper demonstrates how non-isomorphic rotational mappings and interaction techniques can be designed and used to build effective spatial 3D user interfaces. In this paper, we develop a mathematical framework allowing us to design non-isomorphic 3D rotational mappings and techniques, investigate their usability properties, and evaluate their user performance characteristics. The results suggest that non-isomorphic rotational mappings can be an effective tool in building high-quality manipulation dialogs in 3D interfaces, allowing our subjects to accomplish experimental tasks 13% faster without a statistically detectable loss in accuracy. The current paper will help interface designers to use non-isomorphic rotational mappings effectively.

Keywords: 6DOF input devices, interactive 3D rotations, 3D user interfaces, interaction techniques, motor control.

INTRODUCTION

Three-dimensional (3D) computer graphics has advanced from a subject of research curiosity to an indispensable tool in many areas of human activities. While visual quality and rendering efficiency have been rapidly improving, the design of efficient interfaces for 3D applications remains a practical concern for application developers and a vexing problem for researchers in industry and academia [12].

Direct manual control has a very special place in 3D user interfaces design and research: human hands remain the dominant channel of interaction not only for 3D interfaces, but also for traditional 2D graphical user interfaces (GUI) as well for our everyday interaction with the physical world. The quality of interface components that enable users to manipulate objects and scenes in virtual environments has a profound effect on the quality of the whole interface – if the user cannot manipulate effectively, many specific application tasks simply cannot be performed. Consequently, a large amount of research has already addressed various issues in multidimensional manipulation, e.g., designing and evaluating multiple degrees-of-freedom (DOF) input devices, innovating new manipulation interaction techniques, investigating implications of human motor skills on 3D interface design, and many others [e.g. 1, 8, 12, 22, 25].

The design of 3D mappings and interaction techniques, which translate user-operated device motions into object movements in virtual environments, is certainly one of the core issues in designing manipulation interfaces. The chal-

lenge was very well defined by Sheridan (cited from [24]): “How do the geometrical mappings of body and environmental objects, both within the virtual environment and the true one, and relative to each other, contribute to the sense of presence, training, and performance? ... In some cases there may be a need to deviate significantly from strict geometric isomorphism because of hardware limits, or constraints of the human body. At present we do not have design/operating principles for knowing what mapping ... is permissible, and which degrades performance.”

Designing and investigating non-isomorphic mappings for 3D spatial user interfaces has recently been an area of active research [e.g. 2, 8, 12, 13, 14, 15], and the current paper adds to this body of work. In particular, we explore how non-isomorphic rotational mappings can be designed and used to enhance 3D rotations of objects and scenes in virtual worlds. The paper attempts to close a current gap in the literature on multidimensional interaction, where 3D mappings and interaction techniques have been used only with the translation components of multiple DOF input. When it comes to 3D rotations, most researchers, as well as producers of commercial devices and software, have used only the simplest one-to-one (*isomorphic*) mapping between rotations of the multiple DOF controller and virtual objects. In fact, even the basic equations of control-display (C-D) gain for 3D rotations and their properties have not been reported.¹ In comparison, C-D mappings for translation tasks have been used and studied since the early 1940s.

This paper demonstrates how non-isomorphic 3D rotational mappings can be constructed and effectively used to design 3D interfaces. First, we introduce a basic mathematical framework that allows design of both linear and non-linear C-D mappings between device rotations and rotations in 3D interface space. This framework is based on the idea of extrapolating the orientation of a multiple DOF device on a quaternion sphere in four dimensions. Second, we identify basic idiosyncratic properties of rotational mappings, such as relations between the mappings and device form-factor, and discuss issues of interaction techniques design. Finally, we report experiments which have shown that by using our technique, subjects could accomplish an experimental task 13% faster without any significant loss in accuracy.

BACKGROUND AND RELATED WORK

Any interface between humans and machines that uses continuous manual control includes three basic components: 1) input devices, which capture user actions, 2) display de-

¹ We reported preliminary results in the CHI '99 late-breaking paper [16].

vices, which present the effect of these actions back to the user, and 3) transfer functions, often referred to as control-display mappings, which map the movements of the device into the movements of controlled elements of the system or interface [11, 24] (Figure 1). The goal is to design input devices, displays and transfer functions that facilitate high user performance and comfort, while diminishing the impact from human and hardware limitations [11].

The design of mapping functions for manual control and studies of their impact on operator performance stretch back to the 1940s [10]. It has also been an active research area in 3D user interfaces where two philosophies have emerged [24]: The *isomorphic* view suggests a strict geometrical isomorphism (i.e. one-to-one mapping) between motions in the physical and virtual worlds, on the grounds that it is the most natural and therefore is better for users. The results of early human factor studies indicated that while isomorphism is, indeed, often more natural [overview in 11], it also has important shortcomings. First, isomorphic mappings are often impractical because of constraints in the input technologies, e.g., the limited tracking range of input devices. Second, isomorphism is often ineffective due to the limitations of human operators, e.g., anatomical constraints. Finally, it has been argued that 3D interfaces can be more effective, intuitive and richer if, instead of imitating the physical reality, we create mappings and interaction techniques that are specifically tailored to virtual environments, providing in some sense a “better” reality [e.g. 21].

Hence, the *non-isomorphic* approach suggests that manipulation mappings and techniques can significantly deviate from strict realism, providing users with “magic” virtual tools, e.g., laser rays, rubber arms, Voodoo Dolls [13, 14] and others. These non-isomorphic mappings and techniques allow users to manipulate objects quite differently than in the physical world, yet rather effectively [2, 15]. In fact, the majority of 3D direct manipulation techniques today are non-isomorphic techniques.

We should note, however, that non-isomorphic mappings are not an entirely new idea; they have been used for decades in a variety of everyday controls, e.g., dials, pedals, handlers, and wheels, where our input is scaled, shifted or integrated using different mapping or *transfer functions* [11]. Traditionally, the human factors literature categorizes transfer functions by a number of integrations, applied to the user input [11, 24]. Thus, in *zero-order* mappings, displacement of the input device results in displacement of the controlled element, while in *first-order* mappings, it results in a change of its velocity. Consequently, they are often referred to as position and rate control, respectively.

The simplest example of zero-order mapping is a linear control-display gain function which scales the user input:

$$D_d = kD_c, \quad (1)$$

where D_c and D_d are displacements of the controller and displayed elements, respectively, and k is a ratio of scaling. The zero-order control should not necessarily be linear; for example, various dials in consumer electronic devices often use non-linear mappings. Non-linear position control has also been used in VR interaction techniques for object ma-

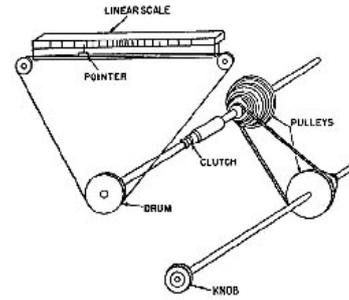


Figure 1: Basic components of any direct manipulation system: input device, output device, and transfer function (in this figure: knob, pointer, and pulleys respectively) [10].

nipulation [14] and navigation [20]. A good example of first-order control is the steering wheel of a car, where the displacement of the steering wheel results in the change of the car’s angular velocity.

The design of non-isomorphic mappings cannot be accomplished without considering the properties of input devices. The most important device property is the number of the degrees of freedom: early research on 3D user interfaces was often concerned with the design and evaluation of techniques for performing 3D tasks with 2D input devices, e.g., ARCBALL or Virtual Trackball techniques, which use a mouse to rotate 3D objects [9, 18]. In multiple DOF input, additional device properties have to be considered. For example, studies by Zhai [24, chapter 2] have shown that isometric devices, such as force-resistant joysticks, allow for better rate control performance, while isotonic devices, such as free-moving magnetic trackers, are preferable for position control. Given that the same device permits a large variety of mappings, device-mappings compatibility is an important and interesting research direction.

The non-isomorphic mappings and interaction techniques have been designed, until now, only for translation components in multiple DOF input. When it comes to 3D rotations, most researchers, as well as producers of commercial devices and software, use only the simplest one-to-one mapping between the 3D rotation of the device and virtual objects. In fact, even the basic equations of mappings that would linearly amplify the device rotations, i.e., linear C-D gain, have not been reported.

What advantages can we gain by using non-isomorphic 3D rotational mappings? Indeed, it can be argued that, unlike translations, the rotation space is limited to 360 degrees, so any desired orientation can be easily achieved. This issue, however, may well be moot. First, the entire 360 degrees of rotations cannot always be tracked; for example, in computer vision-based tracking, the range of rotations that can be reliably measured is often less than 180 degrees [16]. The non-isomorphic mappings would allow a more effective use of this limited tracking range.

Second, the effective range of rotations in manual control is naturally constrained by human anatomy: our joints can only rotate up to a certain angle. Hence, controlling the large range of rotations is difficult and requires *clutching*, i.e., releasing a virtual object, re-adjusting the hand, and continuing the manipulation. Clutching, however, is frus-

trating and can noticeably degrade user performance. While an appropriate device form-factor can reduce clutching [25], it cannot eliminate it. Thus, non-isomorphic mappings can be used to decrease clutching in 3D rotations.

Finally, the introduction of non-isomorphic mappings for 3D rotations would provide interface designers with an additional tool for fine-tuning 3D user interfaces and creating new mappings and 3D interaction techniques.

CONTROL-DISPLAY MAPPINGS IN 3D ROTATIONS

In this section we introduce a basic mathematical framework that allows design of both linear and non-linear C-D mappings between device rotations and rotations in 3D interface space. The design of these mappings is not obvious and requires a consideration of the fundamental mathematical properties of rotations in space. The resulting framework is based on the idea of extrapolating multiple DOF device orientations on a quaternion sphere in 4 dimensions.

Rotations in space

Rotations in 3D space are significantly more confusing than they appear, since they do not follow familiar laws of Euclidean geometry. For example, rotate an object in some direction and it would eventually return to its initial starting orientation, something which cannot happen in a vector space. This is because the space of rotations is not a vector space but a closed and curved surface, a manifold, in four dimensions, which can also be represented as a 4D sphere.

The connection between spatial rotations and spherical geometry is quite natural and can be illustrated using a simple physical example. Imagine rotating a rigid physical object, e.g., a pencil, about a fixed point. Apparently, the tip of the pencil would travel on the surface of a sphere and each *orientation* of the pencil can be identified as a *point* on this sphere. Furthermore, a pencil *rotation* around an axis would draw an *arc* and if the pencil has unit length, then the length of this arc equals the rotation angle. Thus, the orientation of the body can be conveniently represented as a point on a unit sphere, while rotation can be represented as an arc on a sphere, connecting the starting and final body orientations.

This example is illustrative, albeit not quite correct: a point on a 3D sphere specifies a family of rotations, since twisting the pencil along the longest axis would not draw any arcs. Since a sphere in 3D specifies only two degrees of rotational freedom, we need to move into a higher, fourth dimension to specify all three degrees of rotations. This is exactly what *unit quaternions* allow us to do.

Quaternions

Quaternions were discovered by Hamilton in 1843 [17]. Since then, they have been widely used in robotics, avionics and any other application field that requires an efficient way to describe and operate 3D rotations. Introduced into computer graphics and interface design by Shoemake [17, 18], today quaternions are a standard tool in the arsenal of the interactive computer graphics professional.

Quaternion q is a four-dimensional vector often represented as a pair (\mathbf{v}, w) , where w is a real number and \mathbf{v} is a 3D vector. Given quaternion q , we can compute its length $|q|$ and inverse q^{-1} ; given quaternion q' , we can compute their

multiplication qq' and a dot product $q \cdot q'$. A quaternion of unit length can be used to represent a single rotation about unit axis $\hat{\mathbf{u}}$ by angle \mathcal{G} in two equal forms as follows:

$$q = \left(\sin \frac{\mathcal{G}}{2} \hat{\mathbf{u}}, \cos \frac{\mathcal{G}}{2} \right) = e^{\frac{\mathcal{G}}{2} \hat{\mathbf{u}}}$$

Rotating a vector \mathbf{v} about axis $\hat{\mathbf{u}}$ by angle \mathcal{G} can be computed as the double quaternion multiplication $\mathbf{v}' = q\mathbf{v}q^{-1}$. A sequence of rotations q_1, q_2, \dots, q_n can be easily computed as the multiplication $q_n \dots q_2 q_1$ (notice the reversed order; see Appendix for operation definitions).

The set of all unit quaternions forms a unit sphere in four dimensions and each point on its surface represents an orientation of a rigid body. It was proven by Euler that a combination of any number of rotations can be represented as a single rotation from an reference orientation. A unit quaternion represents this single rotation as a *great arc* connecting the reference and current body orientations on quaternion sphere. The length of this arc equals $\frac{1}{2}$ of the rotation angle. Thus, just as we use vectors to represent translations, we also can use spherical arcs to represent 3D rotations. If the reference orientation is not explicitly specified, a quaternion defines the rotation from the identity quaternion $\mathbf{1} = (\vec{0}, 1)$, which has a special meaning as a zero orientation or no rotation – an equivalent to the origin in a vector space.

Linear zero-order C-D gain for 3D rotations

Given an orientation of the multiple DOF input device, what mapping allows us to amplify or scale this orientation in a manner similar to scaling translations of the device?

Amplifying rotation means changing the amplitude while preserving the direction of rotation. Let q_c be the orientation of a multiple DOF input device:

$$q_c = \left(\sin \frac{\mathcal{G}_c}{2} \hat{\mathbf{u}}_c, \cos \frac{\mathcal{G}_c}{2} \right) = e^{\frac{\mathcal{G}_c}{2} \hat{\mathbf{u}}_c},$$

where $\hat{\mathbf{u}}_c$ is the axis of rotation and \mathcal{G}_c is the angle. The zero-order C-D gain should amplify the angle of rotation \mathcal{G}_c by coefficient k while leaving axis $\hat{\mathbf{u}}_c$ intact:

$$q_d = \left(\sin \frac{k\mathcal{G}_c}{2} \hat{\mathbf{u}}_c, \cos \frac{k\mathcal{G}_c}{2} \right) = e^{\frac{k\mathcal{G}_c}{2} \hat{\mathbf{u}}_c} = q_c^k.$$

Therefore, the basic equation for the zero-order linear C-D gain for spatial rotations is a *power* function of the form:

$$q_d = q_c^k, \quad (2)$$

where q_c is the device rotation, q_d is the displayed orientation, and k is the C-D gain coefficient.

Quaternion q_c in Equation 2 specifies device orientation as a rotation from an unspecified initial orientation designated by identity quaternion $\mathbf{1}$. However, it is often important to amplify rotation relative to some explicitly specified reference orientation q_0 . This can be done by calculating the rotation that connects q_0 and q_c , amplifying it, and combining it with reference orientation q_0 :

$$q_d = (q_c q_0^{-1})^k q_0. \quad (3)$$

Notice that Equation 3 is identical to the *slerp* function introduced by Shoemake for rotation interpolation [17]. This should not come as a surprise; indeed, while Shoemake *interpolates* quaternions using a great arc on a quaternion

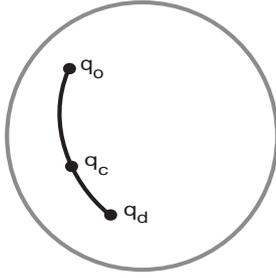


Figure 2: Extrapolating device orientation q_c on a quaternion sphere; q_o and q_d are initial and displayed orientations.

sphere, we *extrapolate* the device orientation using a great arc connecting q_o and q_c (Figure 2)². Therefore, we can use an equivalent formula that is easier to apply [17]:

$$q_d = q_o \frac{\sin((1-k)\Omega)}{\sin(\Omega)} + q_c \frac{\sin(k\Omega)}{\sin(\Omega)}$$

where Ω can be obtained from $\cos\Omega = q_c \cdot q_o$.

Equations 2 and 3 are fundamental equations of rotational C-D gain. They are fundamental in the sense that they represent a basic form of zero-order C-D mappings between rotations of the device and rotations in a 3D interface space and provide a generic method for constructing a variety of rotational techniques suitable for particular application.

These mappings are *linear*, since the rate of amplification does not change no matter how far the user rotates the device. The non-linear mappings, which might be useful, are not discussed here; we refer the interested reader to [16].

INTERACTION TECHNIQUES: DESIGN GUIDELINES

In the previous section, we derived the basic equations of mapping that allow us to linearly amplify rotations of multiple DOF input devices. This section discusses how these equations can be used to design non-isomorphic techniques for rotating objects in VEs. At the center of this discussion are important and non-intuitive differences between absolute and relative mapping schemes in 3D rotations.

Absolute and relative mappings: it makes a difference

Typical isotonic multiple DOF devices, such as magnetic trackers, are absolute devices, i.e., they measure and return the absolute displacement of the device relative to the initial, zero orientation [4]. Hence, the easiest method for implementing non-isomorphic techniques is to map the absolute orientation of device q_{c_i} , measured on i -th cycle of the simulation loop, using Equations 2 or 3:

$$q_d = q_{c_i}^k,$$

and apply the resulting absolute orientation q_d to virtual objects, scenes, and virtual viewpoints.

An alternative way to implement non-isomorphic techniques using the same equations is to amplify only relative changes in the device orientation, i.e. on i -th cycle of the simulation loop, we calculate the relative rotation of device rotation from its orientation on the i -th cycle and amplify

² Spherical arcs have also been used in Arcball [18]. However, using spherical arcs to represent 3D rotations is a standard practice while the purpose and realization of Arcball are different from the present work.

it. The orientation of virtual object q_{d_i} is then calculated by combining this amplified relative rotation with the orientation of virtual object on the $i-1$ step of the simulation loop:

$$q_{d_i} = (q_{c_i} q_{c_{i-1}}^{-1})^k q_{d_{i-1}}. \quad (4)$$

Hence, the difference between these two mapping schemes is that in the first one we amplify the absolute orientation of the device, while in the second one we amplify its relative rotations. Consequently, we will refer to them as *absolute* and *relative* non-isomorphic rotation mappings.

Differentiating between absolute and relative mappings in spatial rotations is important for two reasons. First, they are different from a mathematical point of view: *given the same rotation path of the device, these two mappings produce different rotation paths of the displayed object*³. This might be unexpected; indeed, in the case of translations, relative and absolute mappings would obviously yield the same trajectory of movement. This, however, is yet another example of the peculiar nature of curved rotational space.

Second, *absolute and relative mappings are very different from the usability point of view*. The “feel” of the manipulation largely depends on the choice between relative and absolute mappings. The next section compares and contrasts the usability characteristics of relative and absolute mappings and their implications for 3D interface design.

Usability properties of absolute and relative techniques

Our ability to self-regulate motor movements, e.g., object manipulation, depends on spatial and temporal correspondence between a large variety of sensory feedbacks: visual, tactile, kinesthetic, proprioceptive and others. If the computer response, e.g., visual feedback, conflicts with kinesthetic or proprioceptive feedback produced by the human motor system, then the user performance degrades [19]. Therefore, the effectiveness of manipulation techniques depends on whether they preserve *compliances* between the user motor movements and the sensory feedback s/he receives, i.e., a stimulus-response (S-R) compatibility [6, 19].

In this section, we examine whether absolute and relative rotational mappings preserve two particular compliances which are important for effective direct manipulation: 1) compliance between the rotation directions of the input device and virtual object, i.e., *directional compliance* and 2) compliance between initial orientations of the object and input device, which we refer to as a *nulling compliance*.

Directional compliance

Directional compliance in spatial rotations simply means that as the user rotates the multiple DOF input device, the virtual object rotates in the same direction, i.e., around the same axis. Directional compliance ensures correspondence between visual, kinesthetic, proprioceptive and other feedbacks of motor movement [7, 19]. Britton [3] introduced

³ To prove this, we need to show that if for a sequence of n incremental rotations $q_n q_{n-1} \dots q_1 = q$ then generally $q_n^k q_{n-1}^k \dots q_1^k \neq q^k$ (*). Although an analytical proof is beyond the scope of the paper, it can be easily tested empirically: for $n=3$, $k=2$, $q_1=(0.8,0.6,0,0)$, $q_2=(0.8,0,0.6,0)$ and $q_3=(0.64,-0.48,-0.48,-0.36)$, the left part of equation (*) is $(0.7, 0.3, -0.5, -0.2)$, while the right part is $(0,0,0,1)$.

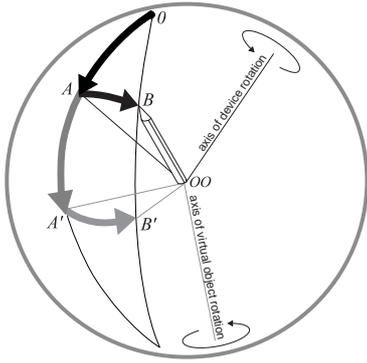


Figure 3: Directional *incompliance* in absolute mapping: (black – actual rotations; gray – amplified rotations).

directional compliance to computer graphics as a principle of kinesthetic correspondence. We can show that:

1) *Relative non-isomorphic mappings always maintain directional compliance* between rotations in physical and virtual spaces. As shown in Equation 4, in each cycle of the simulation loop, the virtual object is incrementally rotated in the same direction as the device, though with a different amplitude. 2) *Absolute non-isomorphic mappings generally do not preserve directional compliance* between rotations of the input device and virtual objects. To illustrate this let us consider a simple physical example. Suppose our input device is a pencil fixed in point OO and it rotates from the initial orientation O to A and then to B (Figure 3). The absolute orientation of the pencil can be represented as an arc connecting initial orientation O and the tip of the pencil. The absolute mapping would always scale this arc *relative to the initial orientation* O , and then apply the resulting amplified orientation to the virtual pencil, which would rotate from O to A' and then to B' . It is obvious from Figure 3 that rotation AB of the physical pencil and rotation $A'B'$ of the virtual pencil will happen around different axes.

Nulling compliance

Nulling compliance ensures that nulling the device, i.e., rotating it into an initial, zero orientation [4], would also rotate the controlled virtual object into a zero orientation. Nulling compliance preserves the consistent correspondence between the origins of the coordinate systems in physical and virtual spaces. We can show that:

1) *Absolute non-isomorphic mappings strictly preserve nulling compliance*. This follows directly from Equation 2; indeed, raising the identity quaternion, i.e., zero rotation, to a power will always yield the identity quaternion. 2) *The relative mappings do not generally preserve nulling compliance*. Nulling the device would not necessarily rotate the virtual object into the expected initial state, but rather into some unpredictable orientation. This follows directly from the discussion in footnote 2.

How important is nulling compliance for direct manipulation in 3D interfaces? The answer depends on the other design variable – the form-factor of the input device.

Non-isomorphic mappings and device form-factor

The shape of a multiple DOF input device can make its manipulation easier or harder, by involving different muscle

groups [25]. The device form-factor can also provide cognitive clues to the user on how it can be used [8]. In addition, we observe that different device form-factors provide different sensory feedbacks on the physical orientation of the device. For example, a device mounted on the user's hand, e.g., a data glove, provides the user with strong kinesthetic and proprioceptive feedback on the device's orientation. The user inherently *feels* the device orientation, hence the inconsistency between the zero orientation of the device and the zero orientation of the virtual object, will be noticed and may degrade user performance.

On the other hand, devices that are not worn on the body but are freely rotated in the user's fingers do not inherently provide any kinesthetic or proprioceptive feedback on their orientation. In cases where the device's geometrical shape can be recognized through tactile feedback, the inconsistencies between orientations of the device and virtual objects can still be perceived. However, a homogeneous form, such as a sphere, provides very little sensory information about its actual physical orientation. For such a device, all orientations are equivalent and it is difficult, if impossible, for the user to discriminate between them. Hence, the nulling compliance becomes unnecessary.

To conclude, multiple DOF devices, which have a spherical form and can be manipulated in the fingers, such as Zhai's Finger Ball [25], are not subjected to nulling compliance constraints. Such devices can be effectively used with non-isomorphic relative mappings, as experimental studies reported later in the paper will demonstrate.

Design trade-off in 3D rotational techniques

The difference between using relative and absolute C-D mappings in designing rotational techniques is a question of a trade-off between the directional and the nulling consistency: we cannot have both (Table 1).

Absolute non-isomorphic mappings can only have a limited use since they do not always preserve the directional com-

	<i>Absolute mapping</i>	<i>Relative mapping</i>
Directional compliance	No, virtual object does <i>not</i> always rotate in the same direction as the device	Yes, virtual object <i>always</i> rotates in the same direction as the device
Nulling compliance	Yes, nulling device <i>always</i> returns virtual object to initial orientation	No, nulling device does <i>not</i> necessarily return virtual object to initial orientation
Device form factor	Worn on body or easy recognizable shape 	Homogeneous, sphere 

Table 1: Design trade-off in 3D rotational techniques

pliance, and therefore do not allow the user to consistently predict the response of the virtual object on the device rotations. These mappings, however, can be useful when the device rotations do not change the axis much. For example, we have used them with satisfactory results for viewpoint control using head rotations tracked by a camera [16].

Relative non-isomorphic mappings can be very efficient in manual control tasks if the multiple DOF input device provides little tactile and kinesthetic feedback on its actual orientation and can be freely rotated in the fingers, such as in the case of a Finger Ball and to a certain degree the Polhemus Space Ball. We will support this observation by presenting the results of the following experimental studies.

EXPERIMENTAL USABILITY STUDY

An experimental usability evaluation was conducted to investigate the performance characteristics of a relative non-isomorphic rotational technique compared with conventional one-to-one mapping in a 3D object rotation task. Based on the results of pilot studies, the following preliminary hypotheses were formulated prior to the experiments.

H₁: A relative amplification of multiple DOF input device rotations will allow subjects to accomplish a rotation task faster than with traditional isomorphic mappings when a large range of rotations is required. A non-isomorphic mapping will not have a significant effect on subject performance for a small range of rotations.

A large range of rotations usually requires clutching or involves larger muscles of the arms and shoulders, and this decreases user performance [25]. We suggest that a non-isomorphic interaction technique with moderate amplification of rotations will allow subjects to use their fingers more effectively, reduce the need for clutching, and therefore result in faster task completion. We hypothesize that non-isomorphic mapping would not result in better performance for small rotations, and we were interested in whether there would be a decrease in user performance.

H₂: Non-isomorphic techniques with moderate amplification of rotations will decrease rotational accuracy.

The higher the sensitivity of a device, the more difficult it is to rotate the device precisely into the required orientation. While it seems logical to assume that accuracy will suffer, we were interested in how significant the decreases in rotational accuracy would be.

Finally, we were interested in estimating subjects' preferences for rotation techniques. The rotation task has an inherently limited range – 360 degrees – and can be accomplished with or without amplification. Strong subject preferences for some of the techniques would indicate that the choice of mapping does make a difference and therefore should be considered in the design of spatial interfaces.

Subjects and apparatus

Twenty unpaid subjects, eighteen male and two female, all right handed, age range from 19 to 35, were recruited from the laboratory subjects pool. None of the subjects had previous experience with 6DOF input devices.

The experiments were conducted in desktop environments using the SGI O2 workstation with a 17" 1280x1024 pixels

true color monitor. The update rate was controlled between 19 and 25 Hz. The Polhemus SpaceBall 6DOF magnetic sensor was selected as the input device, and a mouse was used as the trigger device. The coefficient of amplification in non-isomorphic technique was chosen empirically at 1.8.

Experimental task

The experimental task design followed the design of orientation matching experiments used by Chen [5] and Hinckley [8]. Participants were instructed to rotate a solid shaded 3D model of a house from a randomly generated orientation into a requested, a priori-specified target orientation (Figure 4). The target orientation was a front of the house, indicated by the front door, facing the user. The house model was made to provide maximum clues to understanding its orientation, e.g., asymmetric location of chimney and windows.

The user picked up and released a house by pressing and releasing the mouse button with the non-dominant hand. The user could rotate the house iteratively using clutching – pick, rotate, release, re-adjust the hand, and re-pick the house as many times as necessary to orient it within the threshold of the specified accuracy. When the error of orientation fell below the threshold, which was approximately 18 degrees ($\pi/10$), the house would disappear, cueing subjects that the task had been accomplished successfully. The next trial was presented after a three-second delay.

This task design differed from Chen's in two respects. First, Chen required subjects to rotate a house from a fixed initial orientation into a randomly generated one. We slightly simplified the task by reversing it: the user rotated the house from the initial random orientation into a known one. Second, Chen rated and scored the user's completion accuracy after each task as "Excellent," "Good match," and so on. We used the accuracy threshold instead, because it allowed us to implicitly control the difficulty of the task as well as to provide participants with clear criteria of task completion.

Experiment design and procedure

The repeated measures-within subject experimental design was used. The independent variables were *interaction technique* (one-to-one and non-isomorphic mapping) and *amplitude* of rotation, defined as the shortest rotation required to rotate the house into the target orientation. The amplitude variable had two levels: *small* (a random angle from 20 to 60) and *large* (a random angle from 70 to 180).

The dependent variables were *completion time* and *orientation error*. The completion time was measured from the

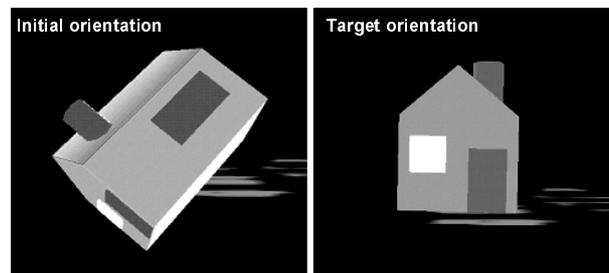


Figure 4: Task required users to rotate house model from randomly generated initial orientation (left) to target orientation, with front of the house facing user (right).

moment the user picked up a house until the moment the house was oriented with the required accuracy. Error was measured as the angular difference between the final orientation of the house and target orientation.

The experiments started with an explanation of the techniques, experimental task and procedure, followed by a 15 to 20 minute training to stabilize the manipulation performance and ensure understanding of the task and techniques. The training was followed by the experimental session consisting of two blocks of trials, one with one-to-one and the other with non-isomorphic mapping. Each block consisted of ten trials: five with large and five with small amplitude of rotation, randomized. All subjects matched the same randomly generated orientations. To control for order effect, half of the subjects started with one-to-one mapping while the other half started with the non-isomorphic technique. In a questionnaire administered after completion of the experiments, subjects were asked to rate the techniques on a scale from 0 to 4 (0 = very bad, 1 = bad, 2 = OK, 3 = good, and 4 = excellent) and explain their choices. The experiments took from 45 minutes to 1 hour for each subject.

Results

A repeated-measures two-way analysis of variance (ANOVA) was performed for each of the dependent variables with *interaction techniques* and *amplitude* as independent variables. Data for completion time was transformed using a natural logarithm, since analysis revealed that the data was skewed away from a normal distribution.

Table 2 outlines the main effects of independent variables as well as their interaction for each dependent variable. Both technique and amplitude significantly affected the completion time. The interaction technique, however, was not a significant factor for the orientation error. A significant interaction between technique and amplitude for completion time suggests that the effect of the interaction technique depends on rotational amplitudes.

A separate comparison of techniques for small and large rotations shows that the non-isomorphic mapping was on average 13.4 percent faster when a large amplitude was required ($F_{1,19} = 7.3$, $p < 0.01$, Figure 5), while no significant difference was found for small rotations ($F_{1,19} = 0.03$, $p < 0.87$). This finding supports the first hypothesis. Both techniques resulted in almost the same orientation error: the average was 6.9 and 6.6 degrees for non-isomorphic and one-to-one mappings, respectively (Figure 5). This difference is insignificant both statistically and qualitatively.

Subjects preferences

In the questionnaire 18 subjects (95%) preferred non-isomorphic to one-to-one mapping; on average they were rated

	<i>technique</i>	<i>amplitude</i>	<i>interaction</i> <i>t*a</i>
Time	$F_{1,19} = 5.52$, $p < 0.03$	$F_{1,19} = 113.932$, $p < 0.0001$	$F_{1,19} = 5.68$, $p < 0.028$
Error	$F_{1,19} = 2.29$, $p < 0.15$	$F_{1,19} = 15.7$, $p < 0.001$	$F_{1,19} = 0.026$, $p < 0.874$

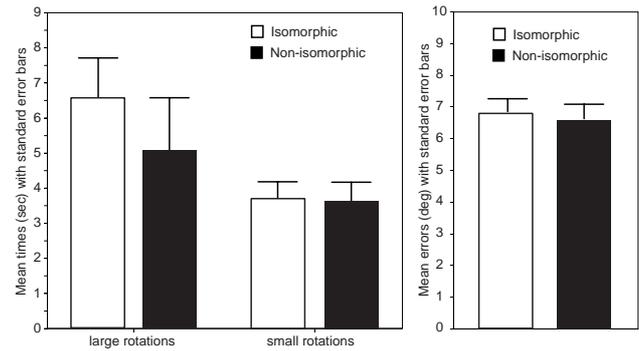


Figure 5: Left: mean completion times; right: mean errors of orientation, collapsed for all amplitudes

3.15 and 2.3, respectively, on a scale from 0 to 4. A paired t -test confirmed that this difference was statistically significant: $t_{19} = 3.9$, $p < 0.001$. Subjects noted that non-isomorphic mappings allowed them to rotate objects faster, with little re-adjustment of hand or device and less physical effort when a large range of rotations was required. Three subjects specifically commented that amplified rotations allowed them to use their fingers over a larger range of rotations, which they found was more efficient.

Many subjects reported that it was slightly more difficult to precisely control device rotations with non-isomorphic mapping. However, many of them noted that it was a question of experience and practice. Two subjects suggested that it would be useful to be able to control the sensitivity of mapping and use slower rotations, especially when accuracy was important. The cable of the SpaceBall tracker was found to be the most disturbing factor in the experiments.

Discussion

The experiments demonstrated that a non-isomorphic interaction technique, which linearly amplifies rotations of a multiple DOF input device, allowed the subjects to accomplish the experimental task 13% faster compared with one-to-one mapping for a large range of rotations. The performance for a small range of rotations was the same. The subjects' strong preferences for the non-isomorphic techniques also suggest that rotational mappings are an important design variable in constructing 3D user interfaces.

Furthermore, mappings had no effect on the accuracy of rotation, or at least none that could be detected with 20 subjects. If we compare our results with Hinkley's experiments [8], who used a similar experimental design, our subjects averaged 6.8 degrees of error, while Hinkley's 24 subjects averaged 6.7 degrees of error. Although our experimental design emphasized speed, our results closely replicated Hinkley's, even though his experiments emphasized accuracy instead. This supports Hinkley's observation that the accuracy of rotation might be less affected by the manipulation capabilities of the interface than by the difficulties subjects had in perceiving and adjusting the rotation error. Furthermore, the experiments of Ware and Rose [23] demonstrated that even when subjects rotated ordinary physical objects in real world, there was a natural limit in accuracy that averaged 4.64 degrees. In 3D interfaces, the accuracy can deteriorate further due to insufficient depth cues or lag.

For completion time, Hinkley's subjects averaged 17.8 seconds while our subjects averaged 5.15 seconds for one-to-one mapping. This difference can be explained, first, by the emphasis of accuracy in Hinkley's experiments, i.e., his subjects spent more time trying to match orientation; second, in the training level, i.e., his subjects were given as little instructions as possible, while we tested the stabilized manipulation performance. The experiments of Ware and Rose, on the other hand, resulted in quite comparable subject performance: their 4.96 versus our 5.15 seconds. Thus, we believe that our experiments produced quite accurate estimates of user rotational performance and accuracy.

CONCLUSIONS

This paper demonstrates how non-isomorphic 3D rotational interaction techniques can be constructed and used to design effective spatial user interfaces. We attempted to provide a thorough treatment of this subject, by designing the mathematical foundations of rotational mappings, investigating their usability properties, and evaluating their user performance characteristics. Our results suggest that non-isomorphic rotational mappings are an effective tool in building high-quality manipulation dialogs in 3D interfaces. This paper will help designers to use them effectively.

ACKNOWLEDGMENTS

Research reported in this paper has been partially conducted as part of the first author's Ph. D. work at Hiroshima University. We are thankful to many people for their suggestions, discussion and help. The basic idea for this work emerged from a discussion between the first author and Jock McKinlay. Michael Svinin and Horst Kraemer have provided invaluable help in discussing the subtle aspects of the mathematics involved in 3D rotations. Takeo Igarashi, Mark Billinghurst, Michael Kowalski and Prof. Ichikawa provided crucial feedback that helped us present these materials in the best possible way. We are also thankful to all subjects who participated in the experiments as well as anonymous reviewers for their comments.

REFERENCES

1. Boritz, J., Booth, K., A study of interactive 3D point location in a computer simulated virtual environment. *Proceedings of VRST'97*. 1997. ACM. pp. 181-187.
2. Bowman, D., Hodges, L., An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. *Proceedings of I3DCG*. 1997. ACM. pp. 35-38.
3. Britton, E., Lipscomb, J., Pique, M., Making nested rotations convenient for the user. *Proceedings of SIGGRAPH'78*. 1978. ACM. pp. 222-227.
4. Buxton, W., There's more to interaction than meets the eye: some issues in manual input. In *User Centered System Design: New Perspectives on Human-Computer Interaction*, D. Norman and S. Draper, Editors. 1986, Lawrence Erlbaum, pp. 319-337.
5. Chen, M., Mountford, S., Sellen, A., A study in interactive 3D rotation using 2-D control devices. *Proceedings of SIGGRAPH'88*. 1988. ACM. pp. 121-129.
6. Fitts, P., Jones, R., Compatibility: spatial characteristics of stimulus and response codes. *Journal of Experimental Psychology*, 1953 (46).
7. Gould, J., Smith, K., Angular displacement of the visual feedback in motion. *Science*, 1962(137): pp. 619-620.

8. Hinkley, K., Tullio, J., Pausch, R., et al., Usability analysis of 3D rotation techniques. *Proc. of ACM UIST'97*. 1997. pp. 1-10.
9. Hultquits, J., A Virtual Trackball. In *Graphics Gems I*. 1990, Academic Press. pp. 462-463.
10. Jenkins, W., Connor, M., Some design factors in making settings on a linear scale. *Journal of Applied Psychology*, 1949. 33(4): pp. 395-409.
11. Knight, J., Manual control and tracking. In *Handbook of human factors*, Salvendy, Ed. 1987, John Wiley&S. pp. 182-218.
12. Mine, M., Brooks, F., Sequin, C., Moving objects in space: exploiting proprioception in virtual-environment interaction. *Proceedings of SIGGRAPH'97*. 1997. ACM. pp. 19-26.
13. Pierce, J., Stearns, B., Pausch, R., Voodoo Dolls: Seamless interaction at the multiple scales in virtual environments. *Proceedings of I3DCG'99*. 1999. ACM. pp. 141-145.
14. Poupyrev, I., Billinghurst, M., Weghorst, S., Ichikawa, T., Go-Go Interaction Technique: Non-Linear Mapping for Direct Manipulation in VR. *Proc. of UIST'96*. 1996. ACM. pp. 79-80.
15. 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*, 1998. 17(3): pp. 41-52.
16. Poupyrev, I., Weghorst, S., Otsuka, T., Ichikawa, T., Amplifying rotations in 3D interfaces. *Proc. of CHI'99 Conference Abstracts, Late Breaking Results*. 1999. ACM. pp. 256-257.
17. Shoemake, K., Animating rotations with quaternion curves. *Proceedings of SIGGRAPH'85*. 1985. ACM. pp. 245-254.
18. Shoemake, K., ARCBALL: a user interface for specifying three-dimensional orientation using a mouse. *Proceedings of Graphics Interface'92*. 1992. pp. 151-156.
19. Smith, T., Smith, K., Feedback-control mechanisms of human behavior. In *Handbook of human factors*, G. Salvendy, Editor. 1987, John Wiley and Sons. pp. 251-293.
20. Song, D., Norman, M., Nonlinear interactive motion control techniques for virtual space navigation. *Proceedings of VRAIS'93*. 1993. IEEE. pp. 111-117.
21. Stoakley, R., Conway, M., Pausch, R., Virtual reality on a WIM: interactive worlds in miniature. *Proceedings of CHI'95*. 1995. pp. 265-272.
22. Wang, Y., MacKenzie, L., Object manipulation in virtual environments: relative size matters. *Proceedings of CHI'99*. 1999. ACM. pp. 48-55.
23. Ware, C., Rose, J., Rotating virtual objects with real handles. *ACM Transactions on Computer-Human Interaction*, 1999. 6(2): pp. 162-180.
24. Zhai, S., Human performance in six degrees of freedom input control, Ph.D. Thesis, *Department of Industrial Engineering*. 1995, University of Toronto, Canada.
25. Zhai, S., Milgram, P., Buxton, W., The influence of muscle groups on performance of multiple degree-of-freedom input. *Proceedings of CHI'96*. 1996. ACM. pp. 308-315.

APPENDIX: QUATERNIONS

The definitions of the quaternion operations used in this paper are as follows:

$$q^* = (-\mathbf{v}, w); |q| = \sqrt{x^2 + y^2 + z^2 + w^2}; q^{-1} = \frac{q^*}{|q|}$$

$$qq' = \mathbf{v} \times \mathbf{v}' + w\mathbf{v}' + w'\mathbf{v}, ww' - \mathbf{v} \cdot \mathbf{v}'; q \cdot q' = \mathbf{v} \cdot \mathbf{v}' + w \cdot w'$$

More information on quaternions as well code samples can be found at <http://www.hitl.washington.edu/people/poup/> or <http://www.mic.atr.co.jp/~poup/>

ErgoDesk: A Framework for Two- and Three-Dimensional Interaction at the ActiveDesk

Andrew S. Forsberg, Joseph J. LaViola Jr., Robert C. Zeleznik

Brown University Site of the NSF Science and Technology Center
for Computer Graphics and Scientific Visualization
PO Box 1910, Providence, RI 02912 USA

1 Abstract

We present ErgoDesk, a software and hardware framework for interacting at an ActiveDesk, a rear-projected drafting table-sized display. ErgoDesk represents a new interaction paradigm, based on physical props, multimodal input and a stereo-on-demand display surface, that seamlessly supports transitions between a variety 2D and 3D interaction techniques. We provide a detailed discussion of the ErgoDesk framework's major features including: the layout of physical props; the transitions between 2D and 3D interactions; the use of gestural and spoken input; and the transformation model for converting between real-world and virtual environment coordinates. We also discuss the design of ErgoSketch, a conceptual 3D modeling application built on the ErgoDesk framework, and our usability experiences with both over the past year.

Keywords: ActiveDesk, Sketch, 3D Interaction, 3D Modeling, Stereoscopic Viewing, Two-handed Interaction

2 Introduction

ErgoSketch is a conceptual modeling system that adapts the Sketch interface [18][19] to the ErgoDesk framework. The ErgoDesk framework is based on the concept of an ActiveDesk (see Figure 1), a variant of the responsive workbench [13], that minimally supports 2D pen-based gestural interaction, 3D interaction, and stereoscopic 3D viewing. The ErgoSketch name derives from the design goal to extend the 2D gestural Sketch application to a more ergonomic physical setup. This setup not only supports natural 2D input, since users draw directly on the display surface with a pen, but also enables seamless transitions between 2D and 3D interaction as dictated by specific interaction tasks.

This paper discusses several aspects of the design of the ErgoDesk framework including:



Figure 1: A user creates 3D geometry at the ActiveDesk with a pen and performs camera operations using the 3D tracker in his non-dominant hand.

- seamless transitions between tools
- 2D pen-based input
- 3D tracked input
- a device layout that supports two-handed interactions
- speech input

Since the design of the ErgoDesk framework is closely tied to the ErgoSketch application, our discussion of the details of ErgoDesk will be given in the context of ErgoSketch. In particular, the the discussion of ErgoDesk will cover specific ErgoSketch tasks including: 3D modeling with 2D gesture lines (as in Sketch [18]), non-dominant hand camera control, stereoscopic model examination in 3D (using an “object-in-hand” metaphor), stereoscopic model annotation in 3D, toolglasses and magic lens interaction [1]. In addition, we will discuss the calibration procedure we are using with our magnetic trackers and how we transform from the ActiveDesk's coordinate system to the virtual environment's coordinate system. The following sections describe

each of these components of the framework in further detail.

3 Seamless Transitions Between Tools

Users switch between a variety of different tools in most modeling applications. It is important to support a seamless transition between these tools to enable an effective interface dialog such that the user may concentrate on the modeling task instead of the tools themselves. We have found that the novel hardware configuration and predominantly gestural input style of ErgoDesk enable more compelling and more seamless transitions than conventional interfaces.

There are several types of transitions in ErgoSketch that appear seamless to the user when contrasted with traditional desktop-style modeling systems. First, transitions between physical tools that support specific tasks are seamless because the tools are positioned on the table top within reaching distance. To switch tools, the user simply puts down one tool and picks up another. Each physical object has a specific purpose just as a pencil, eraser, ruler, and protractor do in traditional drawing. Second, some physical tools serve multiple purposes; in these cases the selection of logical tools is also seamless. For example, a 3D tracker can act as a 3D camera manipulator, a 3D annotation tool, or as a magic lens[1]. The transition between logical tools is accomplished seamlessly by speaking to the tool or automatically based on context. Third, virtual tools are activated and deactivated by a drawn gesture, a spoken command, by selection of a different virtual object, or, less naturally, through a 2D menu selection. A colorpicker, for example, can be activated through selection in a 2D widget-based menu, by drawing a “C” gesture, by speaking “colorpicker”, or by selecting the colorpicker object if it happens to be on the display. Fourth, the system may change aspects of the environment automatically when it perceives certain actions of the user. For example, the world is displayed monoscopically when the user draws 2D lines with a lightpen tool, but when the system recognizes that the user has switched to a tracked 6 DOF proxy tool, the display automatically switches to stereoscopic viewing.

ErgoSketch has a range of 2D and 3D interactions each tailored to best support specific modeling tasks. For example, basic tasks such as object instantiation, object editing, and camera manipulation are accomplished using a lightpen to draw 2D gesture lines that are interpreted by the interface. In addition, a trackball can also be used to manipulate the virtual camera. Other tasks such as virtual object inspection and some forms of object annotation are performed in 3D, closely simulating how such tasks would be performed in the real world. When drawing 2D gestures on a 3D model at the ActiveDesk, a stereoscopic view is problematic because the lightpen can only interact with the display surface. Furthermore, if the 3D model is “above” the display

surface, then the lightpen can break the user’s illusion of the object floating above the desk surface if it is positioned inside or behind the object. Consequently, a monoscopic view is required when drawing 2D gesture lines. However, when interacting in 3D, a stereoscopic view is a more effective way to perceive a 3D model. Therefore, we support a seamless, dynamic transition between monoscopic and stereoscopic viewing depending on which type of tool is being used. In general, environmental transitions are automatically triggered when a particular tool is activated by the user.

3.1 2D Pen-based Input

The Sketch interface [18] is an effective gestural interface for creating and editing 3D conceptual models. Sketch’s first implementation used a conventional workstation setup (a three-button mouse, a monitor, and a keyboard) for input and output. However, the indirect nature of the mouse and monitor display is not ideal for the Sketch interface which interprets “drawn” lines and 2D gestural input to directly manipulate objects and the view.

The ActiveDesk’s drafting table-like design and large display surface augmented with a lightpen naturally supports direct drawing and direct manipulation. There are three buttons on our lightpen: one is triggered when the pen is depressed on the display surface and two others are positioned such that the index finger can press either. We follow the convention of the sketch system where one button is used for drawing gestures, one button is used to manipulate objects, and the third button is used to modify camera parameters (zoom, pan, rotate).

3.2 3D Tracked Input

While sketching in 2D is an effective way to create 3D models, it does not support some inherently 3D tasks such as rapidly viewing an object from different sides. A user might perform this task when examining an object or discussing it with another person. We support this type of interaction through the use of a physical prop that acts as a proxy for a virtual object (similar to [8]). The physical prop is a 6 DOF tracker that normally is attached with Velcro to the edge of the desk surface. When the user picks up the prop, two things happen: first, the object they were designing becomes “attached” to the end of the tracker and interactively translates and rotates as the user manipulates the tracker. Second, the rendering mode dynamically switches to a stereoscopic rendering mode to heighten the illusion of examining a 3D object. In addition to examining the object, the user may use a second 3D tracker to draw 3D virtual annotations around the object.

When the user puts down the tracker that acts as a proxy

for the virtual object, the rendering mode transitions back to a monoscopic display. The user may then pick up the lightpen tool to modify the existing model or create new geometry.

3.3 Device Layout Supporting Two-handed Interaction

There are many situations where two-handed interaction is more effective than one handed interaction. Recent work has demonstrated the benefits of this style of interaction [3][7][14] [16][19]. Our layout of physical devices around the ActiveDesk encourages several types of two-handed interactions. When the user is drawing with the lightpen, the non-dominant hand can manipulate a trackball which can be used to rotate and zoom the camera view with respect to the lightpen. When the user has “picked up” a virtual object with a 3D tracker, the second hand can use a second tracker to create 3D annotations with respect to the first hand— an action that closely mimics similar real world two-handed interactions.

3.4 Speech Input

Speech naturally augments many modeling and general interaction operations. For example, simply saying “colorpicker” can activate a colorpicker widget. In addition, speech can take the place of the keyboard for specifying commands or possibly simplify the task of selecting from a large group of items. Recent advances in speech recognition technology are resulting in robust speaker independent speech recognition for small vocabularies which is nearly adequate for our requirements at this time. We are using a speaker dependent system, [9], in our framework, but are working on integrating a speaker independent system [6]. In most situations, speech is not an exclusive means for performing an operation. The colorpicker widget can also be activated through a drawn gesture (by drawing a “C”).

Spoken commands can be used in the context of operations performed by virtual tools. For example, when the user is holding a tool (e.g., a 3D tracker or the lightpen), he or she can specify additional operations or parameters by speaking. Voice commands are useful in situations where alternative resources are not available. For example, when annotating an object in ErgoSketch both of the user’s hands are occupied (i.e., the non-dominant hand is holding the object and the dominant hand is “drawing” 3D annotations). In this situation, spoken commands provide a way to clear unwanted annotations which previously was done by pushing a button on a keyboard. As another example, the user can issue the voice command “copy” while pointing to an object to duplicate it or speak a color name to change its color.

4 Magnetic Tracker Calibration and Transformations

There are two aspects to the integration of 6 DOF magnetic trackers into our framework for the ActiveDesk. The first is a calibration step which converts *tracker coordinates* into *desk coordinates*. The second is a transform that maps points in the *desk* coordinate system to points in the virtual environment’s (VE’s) coordinate system. This transform is used by the application programmer to easily and intuitively position objects in “3D” in the viewing volume above, on, and below the desk surface. This is especially useful if the application supports general navigation (e.g., translation, orientation, and zooming) through the VE.

The following two subsections describe our calibration procedure and the coordinate transform from desk’s coordinate system to VE’s coordinate system.

4.1 Calibration

We use magnetic tracking technology to perform 3D interactions at the ActiveDesk. To incorporate 3D trackers, it is necessary to calibrate the tracking system because, in general, the tracking system’s coordinate system is not aligned with the ActiveDesk’s coordinate system. However, once calibrated the system does not have to be recalibrated unless the ActiveDesk or the magnetic transmitter is moved.

Figure 2 depicts the six points used to define the ActiveDesk’s coordinate system. To calibrate the desk, the user positions the tracker at each of these six points. Using these samples, a transform (see derivation below) from tracker coordinates to *desk* coordinates is defined. During the calibration, points 1 through 5 are graphically drawn on the desk surface to show the user where to position the tracker. Point 6 is positioned at approximately the user’s eye-level and perpendicular to point 5. Although the height of point 6 above the desk is arbitrary, we have found it convenient to choose an average user’s eye height. We typically use a wooden box as a tool to aid in positioning the tracker perpendicular to the display surface when specifying point 6.

The *desk* coordinate system is defined as follows: the origin (0,0,0) is at the center of the display surface; the *x*-axis points to the *right* and is in the plane of the display surface; the *y*-axis is perpendicular to the *x*-axis and in the plane of the display surface; and the *z*-axis is perpendicular to the *x*-axis and the *y*-axis (according to the right-hand rule). The upper-right, upper-left, lower-left, and lower-right corners of the desk are mapped to the coordinates (1,1,0), (-1,1,0), (-1,-1,0), and (1,-1,0), respectively. Point 6 is mapped to the coordinate (0,0,1). Although unit distances along the *X*, *Y* and *Z* axes are different, this is not an issue for the programmer who typically uses *desk* coordinates only to

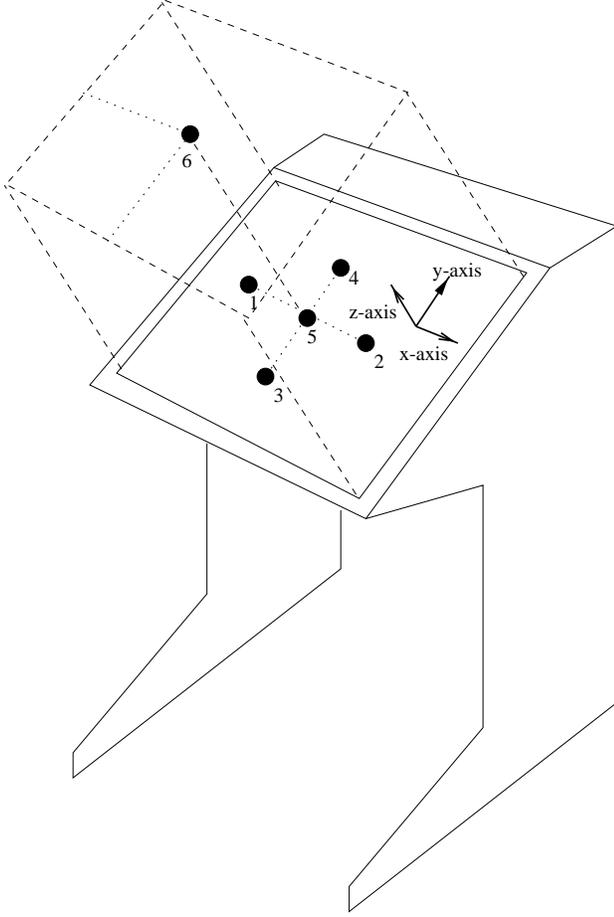


Figure 2: The six points shown are used in the calibration procedure to define the ActiveDesk’s coordinate system. Point 5 is mapped to the origin (0,0,0); the x-axis points in the direction from point 1 to 2; the y-axis points in the direction from point 3 to 4; and the z-axis points in the direction from point 5 to 6.

specify positions relative to the unit cube. Figure 3 illustrates the ActiveDesk’s coordinate system by showing the coordinates of several key 3D points around the viewing area. We now show the derivation of $T_{\text{desk} \leftarrow \text{tracker}}$, the transformation from tracker to desk coordinates.

We define each point n from Figure 2, notated P_n , in terms of a fraction ¹, $\frac{1}{f}$, of the desk surface:

$$P_{1_{\text{desk}}} = \left(-\frac{1}{f}, 0, 0\right) \quad (1)$$

$$P_{2_{\text{desk}}} = \left(\frac{1}{f}, 0, 0\right) \quad (2)$$

$$P_{3_{\text{desk}}} = \left(0, -\frac{1}{f}, 0\right) \quad (3)$$

¹Although this fraction can take any value between zero and one, it is practical to choose a value that is far enough from the origin to minimize the effect of both tracker noise and human placement error, but close enough to the origin that the sampled position is robust (i.e., not on the fringe of a magnetic tracker’s range where there is a lot of noise). f can be any small positive integer—we use a value of 5.

$$P_{4_{\text{desk}}} = \left(0, \frac{1}{f}, 0\right) \quad (4)$$

$$P_{5_{\text{desk}}} = (0, 0, 0) \quad (5)$$

$$P_{6_{\text{desk}}} = (0, 0, 1) \quad (6)$$

We can represent the principal unit vectors of the desk coordinate system (i.e., the vectors (1, 0, 0), (0, 1, 0), and (0, 0, 1)— see Figure 3) in tracker coordinates:

$$\vec{X} = \frac{f}{2} \cdot (P_{2_{\text{tracker}}} - P_{1_{\text{tracker}}}) \quad (7)$$

$$\vec{Y} = \frac{f}{2} \cdot (P_{4_{\text{tracker}}} - P_{3_{\text{tracker}}}) \quad (8)$$

$$\vec{Z} = (P_{6_{\text{tracker}}} - P_{5_{\text{tracker}}}) \quad (9)$$

The transformation from *desk* to *tracker* coordinates is then given by the column matrix:

$$T_{\text{tracker} \leftarrow \text{desk}} = \begin{bmatrix} \vec{X}_x & \vec{Y}_x & \vec{Z}_x & P_{5_{\text{tracker}_x}} \\ \vec{X}_y & \vec{Y}_y & \vec{Z}_y & P_{5_{\text{tracker}_y}} \\ \vec{X}_z & \vec{Y}_z & \vec{Z}_z & P_{5_{\text{tracker}_z}} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (10)$$

Finally, the transformation from *tracker* to *desk* coordinates is the inverse:

$$T_{\text{desk} \leftarrow \text{tracker}} = (T_{\text{tracker} \leftarrow \text{desk}})^{-1} \quad (11)$$

It is interesting to note that in our current system, we do not make any attempt to compensate for static distortion of the magnetic field. However, we have found that the error in the reported samples was essentially undetectable for our application when the magnetic transmitter was positioned about 1.5 feet off the ground, behind the user, and as close to the front of the desk as possible (approximately 2 feet from the front of the ActiveDesk). If greater precision is required, an additional and more involved calibration step is necessary to eliminate static error (i.e., the error in tracked values due to the distortion of the magnetic field due to objects in the physical environment). There are several approaches to reducing static error such as [2].

4.2 Transforming Between Desk and Virtual Environment Coordinates

ErgoSketch uses another transformation between *desk* and VE coordinates to support object placement within the desk viewing volume and to navigate within the VE. By converting between *desk* and VE coordinates it is easy to position

an object relative to the *desk* coordinate system. For example, we can “attach” virtual objects to 3D trackers that report values within the unit cube volume above the desk (see Figure 3) thereby causing the object to appear to float above the desk.

In addition, the *desk* to VE transformation simplifies the programmers interface to moving through the VE. This is similar to the platform concept presented in [17].

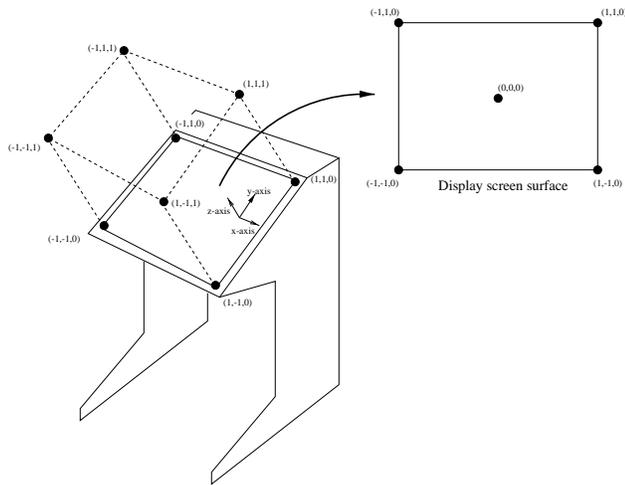


Figure 3: A sampling of points in the ActiveDesk’s coordinate system.

5 Usability Discussion

Over the past few months we have gathered information about the usability of the basic ErgoDesk framework and ErgoSketch modeling application. Although we are primarily interested in the usability of the *framework*, most of our results have been complicated by details of the *application* that uses the framework. In this section we discuss both of these issues in more detail.

In terms of the ErgoDesk framework, we found subjects could easily draw 2D gestures, switch between devices, and transition to visualizing objects stereoscopically. With only a brief explanation it was clear what each tool was for and what effect it could have on the environment. However, subjects had difficulty using the speech recognition which frequently misinterpreted spoken commands, had a limited vocabulary and grammar, was affected by ambient noises, and frequently required the user to carefully enunciate in a constant tone of voice. Some subjects did not like the fact that some devices were fixed or cumbersome to move around in the environment (e.g., the trackball). Further, two-handed interaction was limited to essentially sequential operations (e.g., camera manipulation with the non-dominant hand followed by drawing with the dominant

hand). 2D drawing was hindered by the use of a lightpen instead of a cordless device— although cordless solutions exist from ITI[10]. Finally, subjects expected to be able to use their hands and fingers to *directly* interact with objects on the display surface.

In terms of the ErgoSketch modeling application, users had difficulty creating interesting 3D models. This is in part due to the nature of learning a gestural user interface. The underlying modeling functionality was also relatively incomplete compared to full-featured commercial modeling systems. Lastly, users were strongly affected by the many deficiencies in the underlying hardware (i.e., display blurriness, the tethered lightpen and its noisy data, and difficulty drawing in 3D).

6 Future Work

There are several extensions we would like to add to ErgoSketch that our current framework does not support. We discuss three issues that we will research in order to expand our framework and extend the ErgoSketch system.

At this time, we are not tracking the user’s head position. However, head tracking would result in an improved 3D viewing experience because motion parallax resulting from head motions is a principle means of extracting depth from a scene by the visual system. The user should be able to move their head from side to side when viewing objects in the stereoscopic mode as means for examining objects from different perspectives. While magnetic trackers could be attached to the user’s head in some way, we would prefer to use an unobtrusive camera-based solution. [12] demonstrates such a system for head tracking is robust in various lighting conditions and that no calibration is necessary after the tracking algorithm has begun running. The main advantage of the camera-based solution is that no cabling or method for mounting a tracker on a user is required. The effects of the accuracy of a camera-based head tracker and the sampling rate may be a problem requiring further exploration.

We will also expand the role of speech input in the interface. We are currently supporting simple voice commands. There are situations in which context could further be leveraged and voice commands can be more tightly coupled with individual physical and virtual tools. The QuickSet system [4][11] serves as an example of how speech and drawn gestures can be integrated effectively. In addition, we found most speech commands were spoken in an unnatural way (i.e., overly enunciated) and we believe a much more informal style of communication is necessary.

Lastly, except for 3D annotations, geometry can only be created using 2D gesture lines created with the 2D lightpen. We want to support the generation of both free-form and precise 3D geometry through 3D hand gestures. To

support this functionality, we will add glove devices or camera-based hand tracking to the existing framework and develop interaction techniques to create geometry from 3D hand gestures. The interface for these operations will most likely be multi-modal interactions incorporating speech to modify hand gestures or state the type of geometry to create.

7 Conclusion

We believe ErgoSketch is representative of the next generation of modeling systems which has an interface more appropriate for designers and that reuses learned skills. The ErgoSketch system is built on top of the framework we have developed which enables both 2D and 3D interaction at a projection table and supports seamless transitions between the interactions and tasks.

The prototype system we have developed using the framework has some inadequate implementations. However, many of these problems can be eliminated by purchasing appropriate hardware. For example, a large untethered see-through tablet in place of lightpen technology is less obtrusive way to draw on the desk surface and eliminates the need to calibrate the lightpen. In addition, tablet technology, in theory, supports the tracking of multiple devices that can have unique identifiers which would support a range of interaction possibilities such as personal pens, "Pick-and-Drop" user interface (see [15]), and simplifying the implementation and use of specialized physical tools.

In summary, while the ErgoSketch application was difficult for many users to operate, the underlying ErgoDesk framework appears to be more promising. We believe that many applications can benefit from the ErgoDesk's integration of 2D gestural input, multiple physical tools, transitions between 2D and 3D interactions as well as display modes, and speech input.

8 Acknowledgements

This work is supported in part by the NSF Graphics and Visualization Center, Alias/Wavefront, Autodesk, Microsoft, Sun Microsystems, and TACO. We especially thank Bill Buxton of Alias/Wavefront for donating the ActiveDesk to our research group.

9 References

- [1] Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T., "Toolglass and Magic Lenses: The See-through Interface", *In Proceedings of SIGGRAPH '93*, pp. 73-80, August, 1993.
- [2] Bryson, S., "Measurement and Calibration of Static Error for Three-Dimensional Electromagnetic Trackers", *SPIE Conference on Stereoscopic Displays and Applications*, pp. 244-255, San Jose, CA, 1992.
- [3] Buxton, W., and Myers, B. A., "A Study in Two-Handed Input." *In Proceedings of CHI'86 Human Factors in Computing Systems*, pp. 321-326, 1986.
- [4] Cohen, P. R., Johnston, M., McGee, D., Oviatt, S., Pittman, J., Smith I., Chen, L., and Clow, J., "Quick-Set: Multimodal Interaction for Distributed Applications", *In Proceedings of the 5th ACM International Multimedia Conference*, Seattle, WA, pp. 31-40, November, 1997.
- [5] Forsberg, A. S., LaViola, J. J., Markosian, L., Zeleznik, R. C. (1997), "Seamless Interaction in Virtual Reality", *IEEE Computer Graphics and Applications*, 17(6), pp. 6-9, November/December 1997.
- [6] Hark Recognizer Reference Manual, BBN, Inc. Cambridge, MA 1994.
- [7] Hinkley, K., Pausch, R., Proffitt, D., Patten, J., and Kassell, N. "Cooperative Bimanual Action." *In Proceedings of CHI'97 Human Factors in Computing Systems*, pp. 27-34, 1997.
- [8] Hinkley, K., Tullio, J., Pausch, R., Kassell, N. "Usability Analysis of 3D Rotation Techniques." *In Proceedings of User Interface Software and Technology '97*, pp. 1-10, 1997.
- [9] The In-Cube User Guide, available from Command Corporation, Inc., Atlanta, GA, 1997.
- [10] Input Technologies, Incorporated., <http://www.iti-world.com>.
- [11] Johnston, M., Cohen, P. R., McGee D., Oviatt, S. L., Pittman, J. A., Smith I., "Unification-based Multimodal Integration", *35th Annual Meeting of the Assoc. for Computational Linguistics and 8th Conference of the European Chapter of the Assoc. for Computational Linguistics*, Madrid, Spain, July 7-12, 1997.
- [12] Khuner, H., Personal communication and in print.
- [13] Krueger, W. and Frohlich, B., "The Responsive Workbench", *IEEE Computer Graphics and Applications*, pp. 12-15, 1994.
- [14] Mapes, D. J., and Moshell, M. J., "A Two-Handed Interface for Object Manipulation in Virtual Environments." *PRESENSE Teleoperators and Virtual Environments*, 4(4), pp. 403-416, Fall 1995.
- [15] Rekimoto, J., "Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments", *In Proceedings of User Interface Software and Technology '97*, pp. 31-39, Banff, 1997.

- [16] Shaw, C. and Green M., "THRED: A Two-Handed Design System.", *In Multimedia Systems Journal*, Volume 5, Number 2, ACM/Springer Verlag, 1997.
- [17] Sowizral, H., Rushforth, K., Deering, M., *The Java 3D API Specification*, Addison Wesley Longman, Inc.: Reading, MA, 1998.
- [18] Zeleznik, R.C., Herndon, K., Hughes, J. "Sketch: An Interface for Sketching 3D Scenes." *In proceedings of SIGGRAPH'96*, pp. 163-170, 1996.
- [19] Zeleznik, R.C., Forsberg, A.S., and Strauss, P.S., "Two Pointer Input for 3D Interaction." *In Proceedings of the 1997 Symposium on Interactive 3D Graphics*, Providence, RI, April, 1997.

Design and Evaluation of Menu Systems for Immersive Virtual Environments

Doug A. Bowman and Chadwick A. Wingrave
Department of Computer Science (0106)
Virginia Tech
Blacksburg, VA 24061 USA
{bowman, cwingrav}@vt.edu

Abstract

Interfaces for system control tasks in virtual environments (VEs) have not been extensively studied. This paper focuses on various types of menu systems to be used in such environments. We describe the design of the TULIP menu, a menu system using Pinch Gloves™, and compare it to two common alternatives: floating menus and pen and tablet menus. These three menus were compared in an empirical evaluation. The pen and tablet menu was found to be significantly faster, while users had a preference for TULIP. Subjective discomfort levels were also higher with the floating menus and pen and tablet.

1. Introduction and Motivation

The user interaction in many virtual environment (VE) systems can be characterized in terms of four universal interaction tasks [1]. *Navigation* refers to the task of moving one's viewpoint through an environment, and is divided into a cognitive component (wayfinding) and a motor component (travel). *Selection* is the task of choosing one or more objects from a set. It is often paired with the third task, *manipulation*, which refers to the specification of object properties such as position and orientation. The final universal task, *system control*, can be defined as changing the system state or the mode of interaction. Although travel [2], wayfinding [3], and selection/manipulation [4, 5] have been studied extensively using empirical evaluations, very little research has been done on system control tasks.

One of the most common system control interfaces is the menu. Menus are used to issue commands, begin dialog sequences, change the mode of interaction, and so on. Of course, menus are extremely common in 2D graphical user interfaces (GUIs), and take many forms, including pull-down, pop-up, palette-based, pie, and context-sensitive menus. But are menu systems appropriate for VEs? It is often asserted that all VE

interaction should be “natural,” and divorced from the WIMP (Windows, Icons, Menus, Pointer) metaphor [6]. We take the view, however, that the naturalism of the interface should be based on the application, tasks, and user goals. In a training application, where the goal is to replicate the real world to the greatest possible degree, natural interaction metaphors are preferable. However, in applications where the main user goals are efficient and effective completion of tasks, the interface should be constructed so as to minimize time and errors – such an interface may not be natural.

Moreover, many of the complex application domains for which VE tools have been proposed need to make extensive use of system control. Consider a VE for architectural design in which the user can not only view a 3D structure interactively, but can also modify the space and create new elements [7, 8]. Such a system needs techniques for loading new models, changing an object's texture, and saving one's work. None of these tasks correspond directly to any real-world action, but the tasks could be force-fitted into pseudo-realistic metaphors. For example, the user could enter a virtual library to choose a new object to be loaded into the environment. However, such metaphors are often cumbersome and unnecessary when simpler techniques such as graphical menus or voice commands would be more efficient and precise. Other VE applications such as scientific visualization [9] or science education [10] have similar properties.

Therefore, system control interfaces, and menus in particular, need to be systematically designed and evaluated to ensure high usability and performance levels in VEs. We have experience in the design and implementation of menus for various VE applications, and have used this experience to design and run an empirical evaluation comparing three types of VE menus. These include two previously published menu systems, and a novel implementation of menus using Fakespace Pinch Gloves™. In the next section, we describe related work on VE menu systems and the use of Pinch Gloves™ in VEs. Next, we present the design of the three menu systems tested in our experiment, with particular emphasis on the

iterative design of the Pinch Glove™ menu. We then describe the implementation and results of our experiment.

2. Related Work

Some of the first menus to be used in VE systems were pull-down menus translated more or less directly from their 2D counterparts [11]. These menus floated in 3D space, and were activated via a ray-casting selection technique. The Conceptual Design Space application [8] extended this idea by fixing the menus to the user's head so that they were always in view, and allowing submenus to be used. These "floating menus" were quite usable for experts, but were difficult to learn for some users due to the 3D selection technique. Also, ray-casting was often not precise enough to allow for effective use of submenus, and the menus could obscure a large portion of the environment if they had many entries.

Another idea, the "pen and tablet" technique, is to place 2D interface components on a tracked physical surface within the VE, and allow the user to interact with these components using a tracked stylus [12, 13]. The pen and tablet technique can be used not only for menus, but also for other 2D interface elements such as buttons, sliders, and icons that can be dragged. It has the advantages of a physical surface to act as a constraint, the ability to put the interface away when not needed, and the strong associations with familiar 2D GUIs. We have used pen and tablet interfaces in several VE applications [e.g. 14].

A third type of menus that has been used in VEs takes advantage of the fact that menu selection is essentially a one degree-of-freedom (DOF) operation to provide an important constraint. In these 1 DOF menus, the user controls only one parameter, such as wrist rotation about a single axis, to place the desired menu item in a selection box [7, 15]. These menus can be fast and accurate, but performance suffers when the number of commands becomes large. Moreover, because they use only 1 DOF, there is no notion of a menu hierarchy; instead, the menu is simply a list of commands.

Finally, Mine [16] has explored body-centered menus, in which the menu items are fixed to the user's body (not the head). This allows users to take advantage of their proprioceptive sense when selecting menu items or tools, since they always reside at the same location relative to the body. Theoretically, body-centered menus can be used in an "eyes-off" manner once these locations are learned. Of course, precision is also an issue here when there are a large number of menu items. Body-centered menus also do not inherently support a hierarchy of menu items.

In this paper, we discuss a new menu system design based on Fakespace Pinch Gloves™. Although we know of no published work in which Pinch Gloves™ were used for menus, they have been used for other purposes within VEs. Pierce [17] uses the gloves for novel selection and

manipulation techniques, using different fingers for different functions. The PolyShop system [18] also used gloves for various operations, including viewpoint movement, object placement and scaling, and command selection (with a menu similar to the floating menus described above, where the gloves were simply used to touch the desired selection). Finally, LaViola [19] has prototyped a pair of gloves that combine both pinch inputs and continuous bend sensors to be used for advanced interaction techniques.

3. Designing a Pinch Glove™ Menu System

3.1 Basic Concept

Pinch Gloves™ are a commercial input device designed for use in VEs, in particular on workbench display devices. They consist of a flexible cloth glove augmented with conductive cloth sewn into the tips of each of the fingers (figure 1). When two or more pieces of conductive cloth come into contact with one another, a signal is sent back to the host computer indicating which fingers are being "pinched" (the term pinch is used since the most common gesture involves the thumb touching one of the fingers on the same hand). In terms of logical input devices, Pinch Gloves™ are simply a choice device with a very large number of possible choices. The gloves also have velcro on the back of the hand so that a position tracker can be mounted there.



Figure 1. User wearing Pinch Gloves™.

We felt that an ideal use for Pinch Gloves™ would be for the implementation of a menu system. When designing such a system, several general requirements should be met. First, the new system needs to be at least as efficient and precise as other menu types – performance should not suffer. Second, its use should not cause the user significant discomfort. Third, it should not occlude the environment. Fourth, the menu system should be appropriate for both novice and expert users. Fifth, expert users should be able to do "eyes-off" interaction with the menu.

The gloves allow an almost unlimited number of different gestures. However, simply assigning each menu

item to a different pinch gesture does not produce a usable menu system, since the user has to remember which gesture corresponds to each command. Furthermore, there are very few gestures that have a natural mapping (e.g. pinching the thumb and forefinger represents “OK” in some cultures).

Therefore, something simpler needed to be done. We decided to drastically limit the number of pinch gestures that would be meaningful by using only those gestures in which the thumb touches a single finger on the same hand (eight possible gestures). We also agreed that the menu items needed to be visible to the user so that he is not required to rely on memory. Finally, it seemed important that the menu should allow for some hierarchy or organization of menu items rather than a flat structure containing all commands.

Based on these ideas, we developed the basic concept for our Pinch Glove™ menu system: the top level of the menu hierarchy (menu titles) are displayed on the fingers of the non-dominant hand, and a menu is chosen by pinching the thumb to the appropriate finger; and the second level of the hierarchy (items within each menu) is displayed on the fingers of the dominant hand, and an item is chosen by pinching the thumb of this hand to the appropriate finger. The hands are tracked so that the user can view the labels by moving his hands into view. This design roughly corresponds to research on the use of the two hands [20] stating that the non-dominant hand generally performs coarse, high-level activities while the dominant hand does more precise tasks. In this implementation, both hands are doing the same thing *mechanically*, but conceptually the user is making a less precise selection with the non-dominant hand.

3.2 Initial Prototypes

The main problem with the basic concept for our Pinch Glove™ menu system is that it only allows four top-level menus with up to four items each. Of course, many systems will require more menus and items. Thus, our main hurdle was to find ways to implement longer menus without sacrificing the advantages of the basic design.

Our first prototype solved the long menu problem by placing all menu options on a scrolling list displayed on the palm of the dominant hand. The fingers were then used to scroll up or down in the menu or to select the currently highlighted item (figure 2). This design allows all options to be displayed at once, but may require a relatively large number of pinches (six in the worst case for our test scenario) to select an item.

The second prototype adhered to the basic design more faithfully by allowing the fingers of the dominant hand to select menu items directly. Here, in order to accommodate longer menus, the pinky finger was always reserved for a “more options” item (figure 3). When the menu was

originally selected, the first three items would appear on the first three fingers. Pinching the thumb to the pinky (selecting “more options”) caused the next three options to appear on the other three fingers. In this way, the user could step through sets of available entries until the desired entry appeared, then select it. In the worst case for our menus, four pinches would be required to select an item using this “three-up” design. However, this design does not allow the user to see all the options simultaneously, which might confuse novice users.

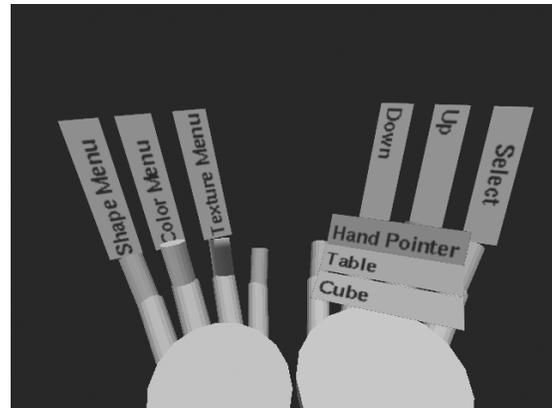


Figure 2. Scrolling menu prototype.

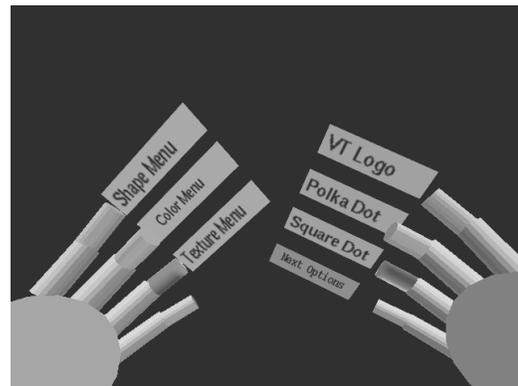


Figure 3. Three-up menu prototype.

3.3 Formative Evaluation

We ran a pilot study in order to evaluate these initial designs. Users wore a Virtual Research V8 head-mounted display (HMD) and the head and both hands were tracked using a Polhemus Fastrak tracking system. We used an HMD even though the task could have been done using a standard monitor because we wanted to simulate as realistically as possible the use of these menu systems in an immersive VE. The test environment was developed using the Simple Virtual Environment library [21].

For testing purposes, we developed a task in which the user changes a virtual object to match a target object.

Three parameters could be controlled: the object's shape, color, and texture (this could be considered a manipulation task, but since it's implemented using menus, we consider it to be system control). Each of these corresponded to a top-level menu. There were three shapes to choose from, eight colors, and six textures – these corresponded to second-level menu items. Thus, the non-dominant hand could choose the top-level menu directly, but support for longer menus was required on the dominant hand. This environment allowed us to measure or observe information relating to each of our requirements.

Four users, representing both novices and those already familiar with VE technology, performed the object-matching task several times using both the scrolling and three-up menus. We alternated the order of presentation so that familiarity with the task or the gloves themselves would not bias the results. We collected informal results through observation of users' actions, a "think aloud" protocol (users were asked to describe their thoughts, goals, and confusions as they used the menus), and a post-evaluation interview.

The primary result of this study was that neither design met all of our requirements. Users preferred the scrolling menu because all of the menu items were visible, but they also realized that they performed the task faster and with fewer pinches using the three-up menu. Both menus initially caused confusion in users: the three-up menu because some choices were not initially visible, and the scrolling menu because users attempted to select items in their palm directly.

Another important finding was that both menu designs could cause fatigue quite quickly, due to holding the hands in front of the face in order to read the labels, and due to holding the hands at an awkward angle in order to read the labels. We experimented with both palm-down and palm-up configurations. Palm-up was more natural to all the users, but the orientation of the labels was not optimal.

We also observed that only one user ever dropped either of their hands out of view. This user was able to remember the order of the menus on the non-dominant hand and therefore could select the menus by feel alone. However, he commented that he probably would not have thought to do this except for the fact that the hand position required to read the labels was uncomfortable!

Finally, several smaller problems were noted, including the lack of feedback when an item was selected, the lack of an indicator as to which menu was currently being viewed on the dominant hand, and difficulty in differentiating the "more options" entry from the other menu entries in the three-up design.

3.4 TULIP Menus

Based on the results of this formative evaluation, we developed a new menu design for the Pinch Gloves™ that

combined the best properties of our two initial designs. The main innovation of this design is the use of the three-up idea to produce better performance while still displaying all of the options for the current menu. As in the three-up design, the first three items in a long menu are displayed on the first three fingers of the dominant hand, and the "more options" item is displayed on the pinky. However, on the palm we also display, in groups of three, the other menu items (figure 4). An arrow connects the "more options" item to the first group of three items, indicating that these three items will become available if "more options" is selected. The groups are arranged in order so that they appear to slide across the hand as each group is made available. We call this the Three-Up, Labels in Palm (TULIP) menu system.

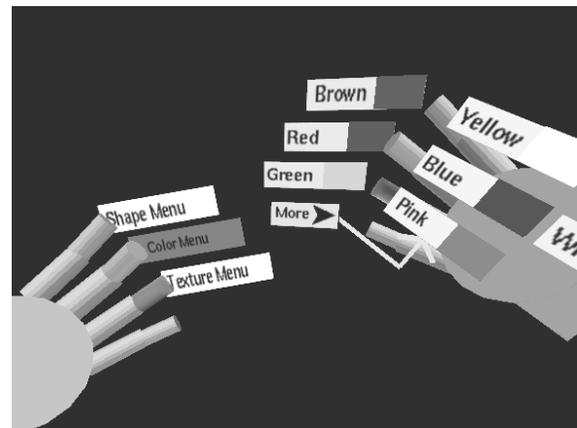


Figure 4. TULIP menus.

We also made several modifications to increase users' comfort when using the menus. First, we moved the virtual representation of the hands 0.25 meters up from the location of the physical hands, so that users could hold the hands at a comfortable level and still see them in the virtual world (interestingly, no subjects in the subsequent experiment noticed this!). Before this modification, it was possible to hold the physical hands in a comfortable position and to see the virtual hands by looking downwards. However, quick downward glances in an HMD are more fatiguing than in the physical world, since head movement (rather than eye movement) must be used. We also rotated the labels on the fingers thirty degrees to allow them to be read while holding the hands at a more comfortable angle. The menu items on the palm were not rotated to help differentiate them from the active items.

TULIP menus also include several smaller improvements. On the non-dominant hand, we indicated which menu was currently active on the dominant hand by a color change on the relevant label. We also provided color feedback on the labels of both hands when a pinch was being performed. The "more options" label was shortened and made a different color to help differentiate it

from the other items, and it disappears if the menu has three or fewer entries.

It might be argued that some of the aspects of the TULIP design would not work well in a more complex VE application. For example, our menus require the use of both hands, so that any pinch between a thumb and a finger on the same hand is interpreted as a menu selection. In real-world VE applications, however, the user is likely to need at least one hand for other operations. This problem could be solved in several ways. First, the system might infer whether menus should be active by hand position: if the user holds his hands palm-up and near to his body, a pinch should be interpreted as a menu selection; otherwise, a pinch should be interpreted as some other type of interaction. Another possible solution is to reserve the non-dominant hand for choosing a top-level menu, but only use the dominant hand for menu item selection when a pinch is held down on the non-dominant hand (in our current implementation, there is always an active menu on the dominant hand). Finally, a special pinch gesture could be used to activate the menu (e.g. touching both thumbs together). This is less desirable since it creates explicit modes, but might work best if both hands were needed for other operations.

Another issue is our decision to modify the position of the virtual hands so that they do not match the position of the physical hands. In a VE application where the hands are used for object manipulation, one might claim this is undesirable. In fact, several published VE manipulation techniques perform some mapping between physical and virtual hand position that is not one-to-one [22, 23] with no apparent ill effects. However, if a one-to-one mapping is clearly desirable, the hand position can again be used to infer the correct action. Hands that are palm-up and close to the body should be raised for comfort in using the menus, and hands at other positions should appear at the correct physical location.

4. Experiment

In order to test our design for TULIP menus, we designed and ran a summative evaluation comparing the glove menu system to two other well-known VE menu types: floating menus and a pen and tablet-based menu. Floating menus, as described above, act like pull-down menus in a desktop GUI. The menu titles are always visible and are attached to the user's head. Selecting a menu title causes the menu items to drop down, from which one can then be selected. We decided to use occlusion selection [24] instead of ray-casting, because our previous experience with floating menus showed that novice users had difficulty selecting the titles using a three-dimensional pointing technique. Our implementation of floating menus is shown in figure 5. The pen and tablet menu system places all menu items onto the surface of a

virtual tablet that corresponds to the surface of a physical piece of cardboard. The virtual pen corresponds to the position of a physical stylus. The menus are separated spatially to show their organization, and the user selects a menu item simply by touching it with the stylus and pressing the stylus button. Our implementation of the pen and tablet menu is shown in figure 6. We did not alter the virtual hand position for these two techniques, since we wanted to compare TULIP to the most common implementation of the techniques.

Our goals for the summative evaluation were to compare the ease of use, ease of learning, efficiency, and comfort of the three menu systems.



Figure 5. Floating menu system.



Figure 6. Physical (left) and virtual (right) views of the pen and tablet menu system.

4.1 Method

To compare these three menu systems, we used the same object-matching task as the formative evaluation. Subjects completed a questionnaire containing demographic information and information about their experience with computers and VEs. They then read a set of instructions discussing the object-matching task in general terms without disclosing the workings of the menu techniques. Before beginning, subjects provided comfort ratings to serve as a baseline for future measurements. The comfort ratings were on a scale of one to ten, with one representing normal conditions and ten representing extreme discomfort, and covered four comfort areas: arm strain, hand strain, dizziness, and nausea.

The equipment and software used was the same as in the formative evaluation, except for the addition of the

Fastrak stylus, used in both the floating and pen and tablet menus, and the tablet itself. The order of presentation of the three techniques was counterbalanced to prevent learning effects from biasing the results. After giving the baseline comfort ratings, subjects were fitted with the HMD and handed the other devices they would use for the first set of trials. Each trial consisted of viewing a target object then changing a second object to match the target’s shape, color, and texture. Subjects completed 30 trials with each menu system. They were instructed to complete each trial as quickly as possible, and were not given any coaching by the experimenter. Thus, subjects learned each interface on their own. After each set of trials, subjects again provided comfort ratings.

We measured the time it took to complete each trial (from the presentation of the target until the match was made) and the number of changes required to make each match. In addition to the comfort ratings for each set of trials, we also polled users about their preferences and perceptions after the experiment was completed.

4.2 Subjects

Twenty-six subjects participated in the experiment. One subject’s data became corrupted, while another subject retired from the experiment, so data was compiled for 24 subjects. The mean age of the subjects was 26.0 years. Four were females. One subject indicated he was left-handed, and another that he was ambidextrous (we did not swap the functions of the two hands for the left-handed subject). Sixteen of the subjects were computer science students, and all but two subjects were students of some type. Half of the subjects had previous VE experience.

4.3 Results

We performed a single factor analysis of variance (ANOVA) using menu type as the independent variable and time per trial as the dependent variable. The average time for trial completion using the TULIP menu was 14.96 seconds; using the floating menus, 12.87 seconds; and using the pen and tablet menu, 11.36 seconds. The ANOVA showed that menu type was indeed a significant factor ($F(2, 23) = 4.23, p < 0.05$).

We also “normalized” the trial times based on the number of required changes, by multiplying the time for the trials requiring only one change by three, and the time for the trials requiring two changes by 3/2 (assuming that there is a fixed time cost associated with each menu selection). The analysis showed in this case that menu type was marginally significant ($F(2, 23) = 2.64, p < 0.1$).

The fact that the pen and tablet menu produced significantly faster performance than the other two menu types should not be surprising. The surface of the tablet allows all menu items to be viewed simultaneously and

selected directly, without hierarchy traversal. By contrast, there are at least two motions (select a menu, select an item) necessary to select a menu entry using floating menus, and between one and four pinches necessary to select a menu entry using the gloves. Moreover, the physical surface of the tablet provides a constraint that prevents users from making mistakes in selection.

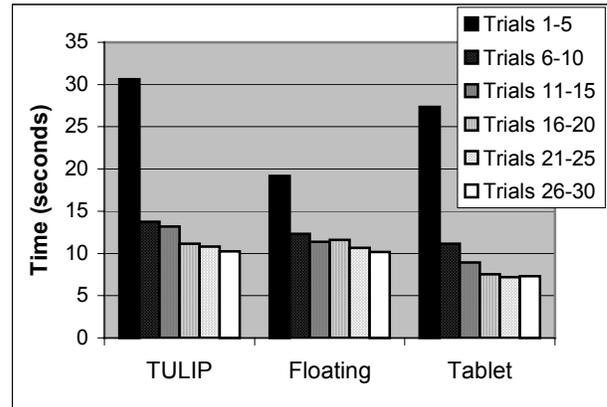


Figure 7. Learning in the menu experiment.

Figure 7 shows the average times for sets of trials with each menu type. We observed that the reason times for the pen and tablet menu are initially poor is that users were not told they needed to look at the tablet in their hand – once they did it was immediately clear what to do.

These results clearly show that learning was taking place with all three menu types. It appears that the gloves were the hardest to learn initially, but performance was at reasonable levels for all three types within five trials. This observation matches the comments we received from many users, such as “the gloves were confusing at first, but once I understood the concept, they were easy to use.” Figure 7 also indicates that performance with the floating menus and the pen and tablet leveled off well before thirty trials, but that users may have still been improving their performance with the TULIP menus. We expect that expert users of the glove-based menus would perform at a level equivalent to or surpassing the other types.

	All Trials		Last 10 Trials	
	VE exp.	No exp.	VE exp.	No exp.
TULIP	12.82	17.10	8.82	12.25
Floating	11.91	13.82	9.81	11.39
Tablet	9.08	13.63	6.61	8.40

Table 1. Average times per trial (seconds) for subjects with prior VE experience and those without.

Table 1 compares performance times between the set of users who had some VE experience and the set of users who had not ever used a VE. It is notable here that for the experienced group, the differences between the menu

systems are smaller, and that in the last ten trials, performance using TULIP menus actually surpassed that of the floating menus for the experienced group. Our criteria for experience was simply that the subject had at some point used a VE system of any kind, so these results indicate that even with a minor level of knowledge, TULIP menus can be quite efficient.

The comfort ratings for each type of menu are shown in figure 8. These results show that floating menus produced a large amount of arm strain (avg. response 5.65), due to the fact that the hand must be held high in the air to use the occlusion selection technique – this replicates an earlier finding [4]. The pen and tablet produced moderate levels of hand strain (avg. response 3.67), although neither of the devices weighs more than a few ounces, possibly due to the lack of a handhold for the user. These results are striking since each menu type was used for no more than ten minutes, and the entire experiment lasted no longer than 45 minutes including rest time. For prolonged usage in an immersive environment, the TULIP menus are clearly preferable.

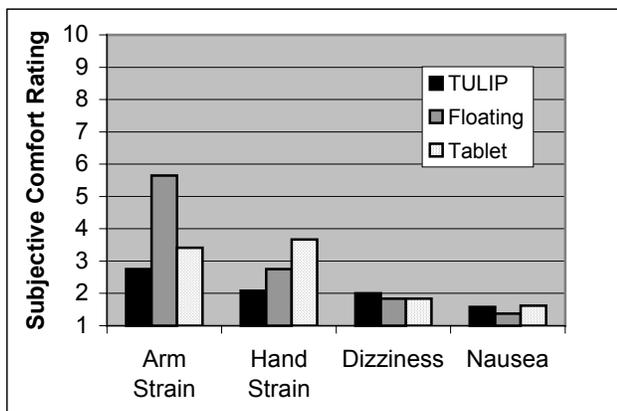


Figure 8. Comfort ratings in the menu experiment.

Fifteen users expressed a preference for the TULIP interface, while nine preferred the pen and tablet, and only two preferred the floating menus (two preferred both the gloves and the pen and tablet). When asked which interface they perceived to be the most efficient, nine users responded pen and tablet, eight responded gloves, and five responded floating menus (two users did not respond).

Combining the efficiency, comfort, and preference information, it appears that both the pen and tablet menu and the TULIP menus performed well in this evaluation. The main drawback of the pen and tablet system is the discomfort it causes users, which might be alleviated by adding an ergonomic handle. The main drawback of TULIP menus is their slightly slower speed, but our subjects did not reach asymptotic performance levels in thirty trials, so expert performance may be equivalent to or better than the pen and tablet system.

4.4 Discussion

Our design for the TULIP menus meets the requirements set forth in section 3.1. Their performance is reasonable, as we have discussed. No significant discomfort was found with the use of the gloves. Occlusion of the environment is still a problem, but is only slightly worse than if the hands were displayed with no labels. The environment is occluded much more by the pen and tablet system, since it uses a single large object. TULIP was more difficult for novice users than for other systems, but even for these users, only a short time was needed to understand the system without any outside coaching. Expert users performed quite well using the gloves. After trying all of the menu systems, a majority of users preferred TULIP. Our last requirement related to “eyes-off” use of the menu system. Although this is possible with the TULIP design, none of the subjects in the experiment did this. We surmise that this was due to the comfortable hand and arm position the gloves allowed, and because occlusion of the environment was not a major problem for the experimental task.

There are other interesting issues with the TULIP menus. As we have seen, performance is worse on the first trials using TULIP. The main reason for this is that the interface is less cognitively direct and has fewer affordances. With both the floating and pen and tablet menus, pressing the stylus button while touching or occluding the proper item causes that item to be selected. With the gloves, labels represent menu entries, but the labels themselves are not directly selected. Rather, the user pinches his thumb to the finger on which the label is seen. This is a subtle distinction, but the lack of directness is enough to cause some confusion in novice users. Users tried many things with the gloves, including pointing at or attempting to grasp the object they wanted to change and selecting the items appearing on the dominant hand with the non-dominant hand.

In addition, users of the TULIP menus cannot reverse an incorrect action because the pinch is the only signal that the item should be selected; whereas in the other two menu types, the user can move the stylus away from the menu item if selection is not desired. On the other hand, once an error is made it may be easier to correct using TULIP because it simply requires another pinch, while the other two menus might require large arm motions.

Overall, this evaluation reiterated some important heuristics from the traditional human-computer interaction literature. Menu systems for VEs need to have good feedback, affordances, and constraints, and items and their actions should be visible [25]. In addition, we found that interacting at a comfortable level, even if this means moving some virtual objects away from their physical counterparts, plays an important role in user performance and preference. TULIP menus were shown to be an

appropriate choice for a wide range of VE systems needing system control interfaces.

5. Conclusions and Future Work

Usable system control interfaces for immersive VEs are essential in order to enable complex and useful VE applications. We have presented the design of a novel menu technique using Pinch Gloves™ and the results of an empirical study comparing it to other common VE menu systems. It is our hope that VE developers will seriously consider these results and their implications when designing menus for immersive environments.

We plan to continue our evaluation of system control techniques. Other menu types should be included, and the experiments should be set in the context of more realistic tasks so that menu usage accurately reflects what might happen in a real-world application.

We also plan to continue to explore the use of Pinch Gloves™ for novel VE interaction techniques. The combination of a large number of possible inputs, the ability to combine the gloves with trackers, and the lack of any object that must be carried makes the gloves potentially useful in a variety of areas, including both natural gestures and abstract uses like the TULIP menus.

Acknowledgments

The authors would like to acknowledge the contributions of Ernst Kruijff, Joe LaViola, and Ivan Poupyrev for their discussions on system control interfaces. We also thank our the subjects in our evaluations for their time and effort.

References

1. Bowman, D. and L. Hodges, *Formalizing the Design, Evaluation, and Application of Interaction Techniques for Immersive Virtual Environments*. The Journal of Visual Languages and Computing, 1999. **10**(1): p. 37-53.
2. Bowman, D.A., D. Koller, and L.F. Hodges. *Travel in Immersive Virtual Environments: an Evaluation of Viewpoint Motion Control Techniques*. In *Virtual Reality Annual International Symposium*. 1997.
3. Darken, R. and H. Cevik. *Map Usage in Virtual Environments: Orientation Issues*. In *IEEE Virtual Reality*. 1999.
4. Bowman, D., D. Johnson, and L. Hodges. *Testbed Evaluation of VE Interaction Techniques*. In *ACM Symposium on Virtual Reality Software and Technology*. 1999.
5. Poupyrev, I., et al. *A Framework and Testbed for Studying Manipulation Techniques for Immersive VR*. In *ACM Symposium on Virtual Reality Software and Technology*. 1997.
6. Nielsen, J., *Noncommand User Interfaces*. Communications of the ACM, 1993. **36**(4): p. 83-99.
7. Mine, M., *ISAAC: A Meta-CAD System for Virtual Environments*. Computer-Aided Design, 1997. **29**(8): p. 547-553.
8. Bowman, D., *Conceptual Design Space: Beyond Walk-through to Immersive Design*, in *Designing Digital Space*, D. Bertol, Editor. 1996, John Wiley & Sons: New York. p. 225-236.
9. Bryson, S., *Virtual Reality in Scientific Visualization*. Communications of the ACM, 1996. **39**(5): p. 62-71.
10. Dede, C., M. Salzman, and R. Loftin. *ScienceSpace: Virtual Realities for Learning Complex and Abstract Scientific Concepts*. In *Virtual Reality Annual International Symposium*. 1996.
11. Jacoby, R. and S. Ellis. *Using Virtual Menus in a Virtual Environment*. In *SPIE: Visual Data Interpretation*. 1992.
12. Angus, I. and H. Sowizral. *Embedding the 2D Interaction Metaphor in a Real 3D Virtual Environment*. In *SPIE, Stereoscopic Displays and Virtual Reality Systems*. 1995.
13. Lindeman, R., J. Sibert, and J. Hahn. *Hand-Held Windows: Towards Effective 2D Interaction in Immersive Virtual Environments*. In *IEEE Virtual Reality*, 1999.
14. Bowman, D., et al., *Designing Animal Habitats Within an Immersive VE*. IEEE Computer Graphics & Applications, 1998. **18**(5): p. 9-13.
15. Liang, J. and M. Green, *JDCAD: A Highly Interactive 3D Modeling System*. Computer & Graphics, 1994. **4**(18): p. 499-506.
16. Mine, M., F. Brooks, and C. Sequin. *Moving Objects in Space: Exploiting Proprioception in Virtual Environment Interaction*. In *ACM SIGGRAPH*. 1997.
17. Pierce, J., B. Stearns, and R. Pausch. *Voodoo Dolls: Seamless Interaction at Multiple Scales in Virtual Environments*. In *ACM Symposium on Interactive 3D Graphics*. 1999.
18. Mapes, D. and J. Moshell, *A Two-Handed Interface for Object Manipulation in Virtual Environments*. Presence: Teleoperators and Virtual Environments, 1995. **4**(4): p. 403-416.
19. LaViola, J. and R. Zeleznik. *Flex and Pinch: A Case Study of Whole-Hand Input Design for Virtual Environment Interaction*. In *International Conference on Computer Graphics and Imaging*. 1999: IASTED.
20. Hinckley, K., et al. *Cooperative Bimanual Action*. In *CHI: Human Factors in Computing Systems*. 1997.
21. Kessler, G., D. Bowman, and L. Hodges, *The Simple Virtual Environment Library: An Extensible Framework for Building VE Applications*. Presence: Teleoperators and Virtual Environments, 2000. **9**(2): p. 187-208.
22. Bowman, D. and L. Hodges. *An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments*. In *ACM Symposium on Interactive 3D Graphics*. 1997.
23. Poupyrev, I., et al. *The Go-Go Interaction Technique: Non-linear Mapping for Direct Manipulation in VR*. In *ACM Symposium on User Interface Software and Technology*. 1996.
24. Pierce, J., et al. *Image Plane Interaction Techniques in 3D Immersive Environments*. In *ACM Symposium on Interactive 3D Graphics*. 1997.
25. Norman, D., *The Design of Everyday Things*. 1990, New York: Doubleday.



PERGAMON

Computers & Graphics 24 (2000) 851–867

COMPUTERS
& GRAPHICS

www.elsevier.com/locate/cag

Calligraphic Interfaces

A multi-layered architecture for sketch-based interaction within virtual environments

Oliver Bimber^{a,*}, L. Miguel Encarnaçãob, André Stork^c

^aFraunhofer Institute for Computer Graphics, Joachim-Jungius-Strasse 11, 18059 Rostock, Germany

^bFraunhofer Center for Research in Computer Graphics (CRCG), 321 S. Main St., Providence, RI 02903, USA

^cFraunhofer Institute for Computer Graphics, Rundeturmstrasse 6, 64283 Darmstadt, Germany

Abstract

In this article, we describe a multi-layered architecture for sketch-based interaction within virtual environments. Our architecture consists of eight hierarchically arranged layers that are described by giving examples of how they are implemented and how they interact. Focusing on table-like projection systems (such as Virtual Tables or Responsive Workbenches) as human-centered output-devices, we show examples of how to integrate parts or all of the architecture into existing domain-specific applications — rather than realizing new general sketch applications — to make sketching an integral part of the next-generation human-computer interface. © 2000 Elsevier Science Ltd. All rights reserved.

Keywords: Sketch-recognition; Sketch-interpretation; Virtual reality; Human-computer interface; Software architecture

1. Introduction

Leonardo da Vinci's drawings of machines and other objects illustrate one of the most fundamental purposes of sketches: the ability to communicate design and functionality to others. Nowadays, it is widely accepted that sketching is a form of critical, reflective dialog that handles communication on one or more different levels of abstraction simultaneously [1]. Various approaches have been taken to support this kind of dialog between humans and computers, and to build human-computer interfaces that are able to interpret such freehand sketches for different purposes.

In this context, the creation or reconstruction of 3D objects from 2D sketches is of major concern in many application areas. This so-called “pencil-and-paper” approach is used for rapidly designing approximate

three-dimensional scenes. While some systems analyze the orthographic or perspective projections to reconstruct 3D shapes that, based on psychological assumptions, are most plausible to the human observer, others interpret 2D gestures while the objects are sketched.

Within the last decade, the conceptual design phase has been increasingly supported by sketch systems that allow the expression of ideas on a computer-aided, but still human-centered basis. However, putting an emphasis on sketching, most of these systems are sealed off from real-world applications rather than being generally applicable as components.

To prevent a separation between sketch systems and real-world applications, we propose the integration of current results into an architectural pattern that offers existing applications an individual utilization of sketching within their user interface. Multi-layered architectural patterns are widely employed in many areas of software and hardware engineering. They offer multiple levels of abstraction, component reuse, exchangeability and encapsulated reengineering of single components, the individual combination of components, and extensibility through pre-defined interfaces.

With this article, we want to introduce a multi-layered architecture for sketch-based interaction within virtual

* Corresponding author. Tel.: + 49-381-4024-110; fax: + 49-381-4024-199.

E-mail addresses: obimber@egd.igd.fhg.de (O. Bimber), menarna@crcg.edu (L. Miguel Encarnaçãob), stork@igd.fhg.de (A. Stork).

environments to benefit from the inherent advantages mentioned above. We describe each layer of the architecture using examples illustrating its implementation. Furthermore, we present a variety of domain-specific applications of sketching within virtual environments based on our architecture instead of implementing yet another application for sketching.

2. Previous work on sketching

Brown University's Sketch system [2] is an example of an early development that processes 2D strokes while they are sketched on the film plane to create predefined 3D primitives.

Since Sketch supports the creation of simple CSG-like primitives, its concept has been extended towards freeform modeling. Teddy [3], for instance, is another desktop-based system that allows the creation of 3D polygonal freeform surfaces from sketched 2D silhouettes.

STILTON [4] is yet another desktop-system that allows the construction of three-dimensional geometry from 2D-perspective (orthographic) straight lines. The system can also be used to create approximated 3D scenes on top of a photograph of a real environment (by drawing over it) that contains minimal geometric information (such as the ground plane) or an existing VRML model.

Sketching three-dimensional scenes on a two-dimensional basis forces the user to artificially mediate the correct view and perspective in terms of giving an impression of the missing third dimension. This causes ambiguity while interpreting geometric properties of the sketched objects, such as type, position, alignment, size, etc.

In Sketch, this problem is solved by using a default parallel projection and by constraining the user. For instance, various aspects of most predefined gestural primitive representations can only be sketched axis-aligned with respect to existing objects and to the current view of the 3D scene. Aligning new objects with existing ones throughout the sketching process is supported whenever possible.

Similar to Sketch, STILTON forces the user to sketch over existing objects in terms of using the alignment information for sketch interpretation. STILTON, however, does not create predefined primitives (as Sketch does) but uses a set of straight-line strokes to construct any geometry. This is achieved by making heuristic assumptions that are represented by a linear combination of objective functions (e.g. face planarity, minimal standard deviation of angles, face alignment, etc.). The combined objective functions are then minimized using genetic algorithms (GAs).

Teddy [3] only allows users to create and modify single objects, which must be topological equivalents to

a sphere. The drawn 2D silhouette of the object is automatically inflated in both the negative and the positive Z-direction to a size that depends on the distance between the neighboring regions on the silhouette. Thus, wide areas become fat and narrow areas become thin. The object can then be modified by using the supported geometry operations (see below).

Besides object creation, freehand-sketches are also used to perform other tasks, such as object selection, interaction and manipulation, and system control. Sketch, for instance, gives users direct interaction with existing objects, indicated by click-and-drag actions. In addition, the transformations can be restricted by sketching corresponding constraints in advance. Teddy allows the application of specific geometry operations (such as cutting, extrusion, smoothing, and mesh transformations) to the freeform objects that are indicated by sketches; scribbled strokes are used to erase objects. Extracted from the underlying photograph, STILTON offers the possibility of automatically mapping texture onto the corresponding geometry.

While the systems described above support strictly two-dimensional desktop environments (e.g. screen, mouse or pen/pad-like devices), large and immersive projection systems — virtual workbenches, virtual walls, surround screen projection systems (e.g. CAVEs or RAVEs), augmented environments, etc. — offer three-dimensional interaction.

Sachs' 3-Draw System [5] is one of the pioneering works that offered sketching directly within the 3D free-space. Supporting two-handed interaction, Sachs used a tracked pad and a tracked stylus to outline object contours with three-dimensional curves (either freeform, constrained, or reflected). The virtual object was attached to the pad, thus the pad served as an interactive sketch-and object palette. In addition to the drawing of curves, the editing and fitting of linked curves was supported. In contrast to later approaches (see below), the 3-Draw system consisted of a non-immersive desktop workspace (a CRT screen) that neither supported head-tracking nor stereoscopic projection.

With ErgoSketch [6], Brown University adapted their Sketch system to an ActiveDesk, a variant of the Responsive Workbench. It supports two-handed interaction tasks, such as 3D modeling with 2D gesture lines (supported by the Sketch interface), non-dominant hand camera control, object-in-hand metaphor, tool glasses [7] and magic-lens interaction [7,31]. Sketching is still performed in a non-stereoscopic (monoscopic 2D) mode on the desk's surface, using a lightpen. However, to interact with the scene in 3D, a stereoscopic (non-head-tracked) mode is automatically activated, triggered by the type of tool being used. One interaction tool is a spatially tracked physical prop that serves as a proxy for virtual objects (i.e. the virtual objects are attached to it and follow its six-degrees-of-freedom (6DOF) motions). In

addition, a trackball is used (e.g. to support camera control), and speech recognition is applied to fire simple commands (e.g. to activate a color-picker).

Surface Drawing is an approach by Schkolne et al. [8] that supports the creation of three-dimensional freeform geometry by hand gestures at a table-like rear-projection display. The user wearing a data-glove can sketch surfaces freely in 3D space. While doing that, the system samples the position data and generates a mesh surface using a fast triangulation scheme.

Forsberg et al. [9] also address 3D curve creation at a table-like display device, supported by sketching. Although there is a variety of related work that has not been mentioned, the referenced systems represent a good state-of-the-art cross-selection and will be used for comparison throughout this article.

3. The architecture

Our architecture (cf. Fig. 1) consists of eight hierarchically arranged layers, which are described below by giving examples of how they are implemented. Each layer can interact with its direct upper or lower neighbor, whereby every layer can be deactivated, making the next activated layer a direct neighbor. This modularization concept offers applications the opportunity for individual utilization of the required functionality.

The core layers (emphasized in Fig. 1) contain the intelligent parts of the architecture, which are being widely implemented by applying methods of artificial intelligence.

Each layer (or single component within the layers) can be updated by alternative and improved versions, thus, an adaptation of the architecture to an ongoing evolution of the components (e.g. caused by technological developments) can be supported.

The limited dependencies between layers and components also allow for distributed processing. In contrast to

a centralized approach, a distributed modality processing, for instance, offers an extensive speedup of the application, as well as the utilization of heterogeneous software and hardware. In our example, both applied modalities (speech and gestures) are independently analyzed on different processors (as described in Sections 3.3 and 3.5), and are merged on a central node (as described in Section 3.6.)

For our discussion, we will assume the architecture on top of the following technology: A workbench-like projection system (a BARCO Baron Virtual Table [10]) serves as output device in our current setup. In addition, we make use of stereoscopic viewing supported by shutter-glasses and head tracking using an electromagnetic tracking-device. Our aim is to make sketching functionality usable for interaction within virtual environments.

3.1. Interaction device

The interaction device represents the lowest level that sits right on top of the hardware-technology. It strongly influences the ‘interaction techniques’ layer above it. However, it is also related to the applied hardware underneath. Featuring two-handed interaction, we use a transparent Plexiglas pad and a pen as interaction devices. The 3D graphics that are projected on the Virtual Table’s display is used to augment the pad [11,12]. Fig. 2 illustrates how to use our setup as a translucent sketchpad [13] to draw two-dimensional freehand-sketches on it.

Both devices are tracked within their 6DOF and can be used separately or in combination. In combination, the pad can provide tactile feedback, while simultaneously taking advantage of the user’s innate ability of knowing precisely where both hands are relative to each other (kinesthetic). A similar setup has been applied by Sachs [5] for the same reasons. However, due to a different projection technology (non-stereo desktop-display), he used an opaque pad. Other opaque pad and pen

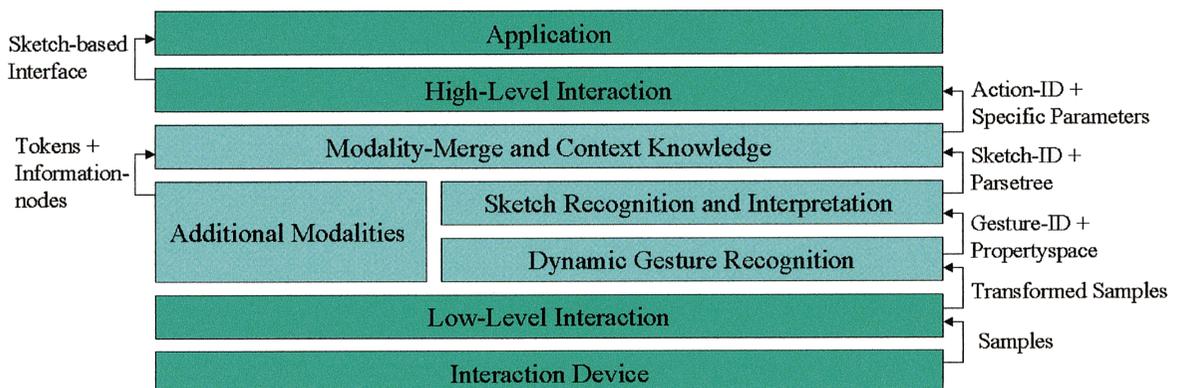


Fig. 1. Multi-layered architecture for sketch-based interaction within virtual environments.

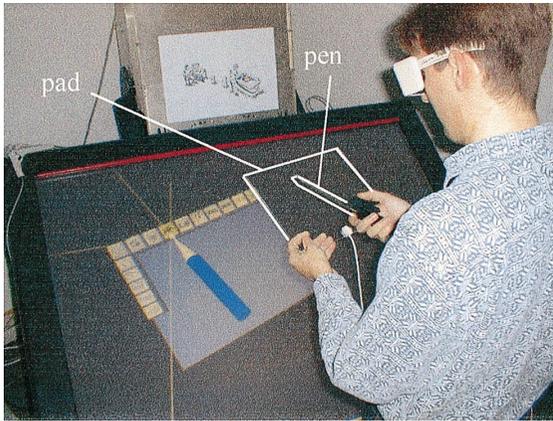


Fig. 2. Translucent sketchpad.

combinations can also be found at some immersive or see-through head-mounted-display (HMD) based systems. The Virtual Notepad [14] is an example of a system that offers handwriting in immersive virtual environments. The personal interaction panel (PIP) [15] is yet another example of an opaque pad and pen that are used with an augmented reality application, called Studierstube [16].

3.2. Low-level interaction

Based on the utilized interaction devices, this layer implements several interaction techniques that support the sketching process within the 3D-freespace. Since two-dimensional or three-dimensional virtual objects and virtual sketches should coexist, we aim at providing equivalent interaction techniques for all of them. Note that the term ‘low-level interaction’ does not mean ‘simple-to-realize interaction’, but refers to the position of this layer within the architecture.

Holding the pen with the dominant hand and the pad with the non-dominant hand offers the user two-handed interaction. In contrast to the approaches of the ErgoDesk [6] to support a seamless transition between 2D and 3D, we offer an embedding of 2D in 3D providing the possibility of constraining the sketching process to the two-dimensional pad area. This has also been realized in other systems, such as the Virtual Notepad [14].

On the one hand, the 2D freehand-sketches are attached to the pad, which, therefore, serves as an interactive sketch-palette [13]. On the other hand, sketches can be drawn directly within the 3D-freespace (similar to Sachs’ 3-Draw [5]). 3D freehand sketches can either be attached to the pad or directly transformed with the pen in the 3D space (Fig. 3).

Objects and sketches that are attached to the local coordinate system of the pad can be intuitively placed at

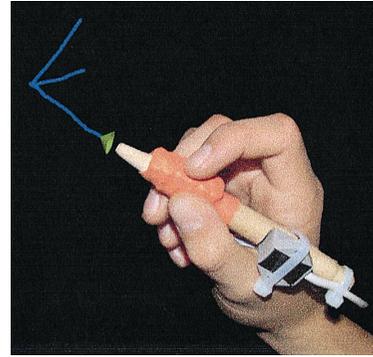


Fig. 3. 3D freehand sketch attached to the pen’s tip.

any position within the global coordinate system of the table via 3D drag-and-drop functionality. For more precise sketching, we apply a stroke-snapping constraint, enabling the virtual representation of the pen (or tip) to snap to any point on any previously drawn stroke. Visualizing virtual representations of the interaction devices (i.e. the pen and the pad) is important to avoid conflicts caused by tracking misalignments (distortion and calibration inaccuracy). In contrast to their real counterparts, the virtual representations do reflect misalignments, thus, they can be taken into account while sketching and interacting.

3.3. Dynamic gesture recognition

The layer ‘dynamic gesture recognition’ is implemented by our interface for motion-based gesture recognition (IMGR).¹

IMGR is implemented as a generic C++ template library and offers the possibility of recognizing single multidimensional motions [17]. (Fig. 4 illustrates the hierarchically layered IMGR concept that is embedded within our dynamic gesture recognition layer.) While providing the necessary set of core functionality, IMGR was designed to be completely configurable and extensible.

Multidimensional stroke samples (by default 2D, 3D or 6DOF) are the basis for the recognition of dynamic gestures. While constrained sketching on the sketchpad, for instance, generates two-dimensional stroke samples, unconstrained 3D-sketches pass 3D samples to IMGR. Although the recognition of 6DOF motions is given, we have not yet implemented an application for this.

The stroke enhancement module offers a palette of smoothing filters (such as one-dimensional Gaussian- or

¹IMGR is available for non-commercial usage. Please contact the authors for further information.

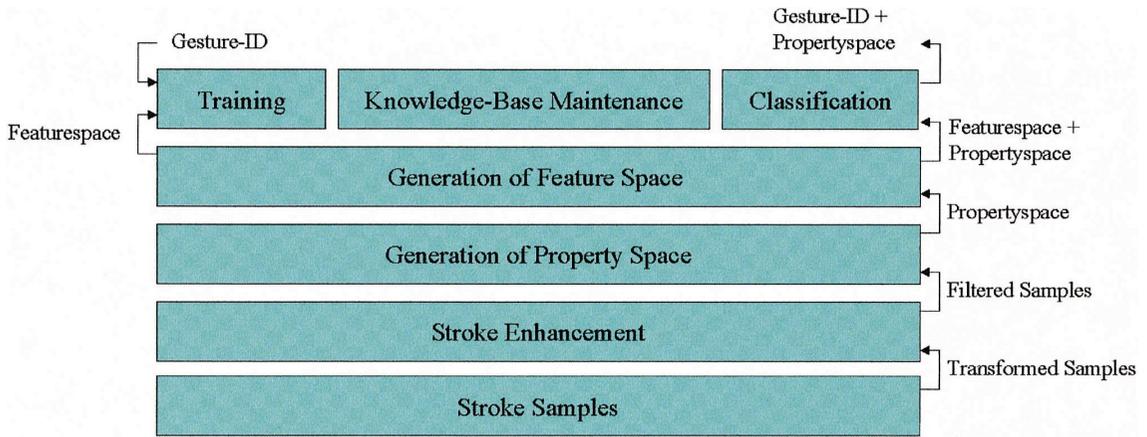


Fig. 4. Multi-layered IMGR concept.

average sliding window kernels) to eliminate peaks and high-frequency sub-bands that are caused by the tracking distortion. This not only enhances and improves the sketching process, but also simplifies classification.

The filtered samples are then used to compute an n -dimensional property space (a list of properties that describe geometric or dynamic characteristics of a stroke). The property space itself can be outlined by defining computation functions for each single property (e.g. geometric or dynamic properties) in advance. In IMGR, however, appropriate property spaces for 2D, 3D and 6DOF gesture recognition have been defined by default and can be extended or refined by defining new computation functions through a standard interface. Examples for geometric properties are center of gravity, total length (see Eq. (1)), axial expansion and axial motions, start and end points, etc. (see [17] for a complete list of implemented properties). Examples for dynamic properties are average velocity or average acceleration.

$$l = \sum_{i=1}^{n-1} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2 + (z_i - z_{i-1})^2} \quad (1)$$

(total length of a gesture stroke with n sample points).

Note, that 38 computation functions of properties have been defined for 3D dynamic gesture recognition in our current implementation.

Since the geometric or dynamic properties are not well suited for being used for classification, an m -dimensional feature space (a list of features that describes geometric or dynamic behaviour of a stroke on a comparable level), is generated. To achieve good confidence values and offer an easy way of describing the classification features, the feature-space is build from fuzzy sets. To do so, we apply the same principle as for the property-space, and outline the feature space by defining membership functions for

the fuzzy sets through a standard interface. As for the property space, appropriate 2D, 3D and 6DOF feature spaces are predefined and can be extended and refined. An example for a feature is its relative straightness (see Eq. (2)). Since a stroke's length is at least as long as the diagonal of its bounding box, Eq. (2) describes the stroke's straightness on a relative (i.e. comparable) basis. Values around 1, for instance, indicate a relative straight progression of the stroke, while values close to 0 indicate a more intricate and complex gesture.

$$rs = \frac{\sqrt{w^2 + h^2 + d^2}}{l} \quad (2)$$

(relative straightness of a gesture stroke).

where w, h, d are the width, height, and depth of the stroke's bounding box.

Further examples for features are bounding-box ratios, relative (to bounding box) start and end points, relative start and end directions, average flexion and overall straightness, etc. (see [17] for a complete list of implemented features). Note, that about 27 computation functions of features have been defined for 3D dynamic gesture recognition in our current implementation.

The reason for differentiating between properties and features is, that the features are much better suited for classification since they can be compared with features of other gestures in terms of determining if they are similar or not. In contrast to properties (that express absolute values), features are defined in a relative (normalized) value-range, but have no further use, while the (mostly geometric) properties can be accessed afterwards by layers above the dynamic gesture recognition (e.g. to support sketch interpretation or high-level interaction).

We generate our knowledge base from a set of feature spaces i.e. a list of comparable gesture-specific comparable characterization criteria. Since we offer different classification methods (that can be extended as well)

such as closest neighbor match (CNM) [17] or linear separation implemented by a feed-forward perceptron neural network [18], which can also be extended or exchanged in order to satisfy different recognition demands. We furthermore support different knowledge-base representations corresponding to the classification methods *accumulated gestures* [17] or *updated class-specific feature vectors* [18].

Online and offline adaptive machine learning is supported for all offered knowledge-base representations. The online training possibility enables users to train the system on demand (e.g. during runtime) and to specify their own gesture sets within the application.

The remaining component contains techniques to maintain and improve the knowledge bases, such as principle component analyses (PCAs), that is implemented by a neural network build from linear associators [19,20] to reduce the contained redundancy.

3.4. Sketch recognition interpretation

In contrast to most of the mentioned systems (Sketch [2], ErgoSketch [6], STILTON [4], Teddy [3], and 3-Draw [5]), we assume that sketches consist of a sequence of dynamic gestures (i.e. freehand strokes) which can be recognized and trained on an adaptive level by the user (i.e. no explicit programming for changing or extending the gestures is necessary), while the application is running.

Since every single gesture (i.e. stroke) is associated with an identification number (generated by the dynamic recognition layer) after classification, we recognize representative multistroke sketches by parsing a sequence of strokes [21]. The sketch representations are defined within a context-free grammar (see Table 1, for example) that is used to generate a parser by applying a standard parser generator (such as Yacc or Bison [22]). The parser that is specific to the sketch language (i.e. the set of supported representative sketches) is automatically integrated into IMGR and is applied to perform sketch recognition on a hierarchical basis (i.e. single strokes are recognized and interpreted by IMGR and the extracted information — such as properties and identification numbers — are passed to the parser).

In addition, we make use of the semantic actions that can be defined for every single production within the grammar to interpret the sketches. Since the generated parser is embedded into IMGR, the semantic actions can access property-spaces that have been generated by the previous layer to perform sketch interpretation.

Every parsed production generates a node of a parse tree that contains the interpreted information (mostly accumulated geometric information) of a sketch component, and passes it (using the build-in parsers stack) to the productions that it is derived from, to combine it with others. Thus, the parsing process assembles a parse tree

whose node information becomes more complete the higher they travel up the parse tree. Once a representative sketch can be recognized (i.e. a root production could be reached), it is hierarchically interpreted and the sketch-specific information is stored in the top node of the parse tree (interpreted information of the sketch components are stored in lower level nodes).

Note that the sketch recognition and interpretation is not adaptive, and appropriate grammars have to be defined in advance. However, they can be dynamically exchanged by IMGR to simultaneously support multiple sketch languages, and since the dynamic gestures at the lowest level are adaptive, single strokes can also be exchanged by the user during runtime.

To support look-ahead, bottom-up parsers as well as users who must learn the representations, we defined sketch languages that assemble higher-level sketches from lower-level ones (Fig. 5 illustrates the 3D-sketch language to create primitives that is shown in listing 1.² The dashed lines represent the composition possibilities of the more complex primitives from less compound ones). The lowest level represents a set of basic gestures, such as points, lines, arcs, circles and freeform strokes.

Elementary strokes (written in capital letters in Table 1) and their pre-computed property spaces are passed from the dynamic gesture recognition layer to the parser. After the stroke-specific property spaces are copied onto the parser stack, the arrival of an elementary stroke triggers the parsing process. Objects are recognized from a sequence of elementary strokes by parsing them through the production rules (lines 4–48 in Table 1) until the root production (line 4) can be derived. During the parsing process, each production determines the object-specific properties within their semantic actions (to maintain the simplicity of Table 1, the semantic actions are denoted as comments between the cambered brackets), copies them onto the parser stack, assembles a new node (that contains the property information) to the parse-tree, and re-triggers the parsing process by returning the derived object type. The semantic action of the root production (line 8 in Table 1) finally passes the recognition results as well as the interpreted property information (i.e. the parse-tree and object properties) to the next layer (via the parser-stack).

Note, that some production rules represent more than one object (e.g. lines 21–23 in Table 1), thus they have to determine the object type in addition. A single freeform stroke, for instance, can outline the contour of a planar freeform-face or the silhouette of a body of revolution. To determine whether a freeform face or a body of revolution has been sketched, the semantic action (indicated in lines 22 and 23) computes the distance between the start point and end point of the freeform stroke and makes the

² Please contact the authors for the complete grammar.

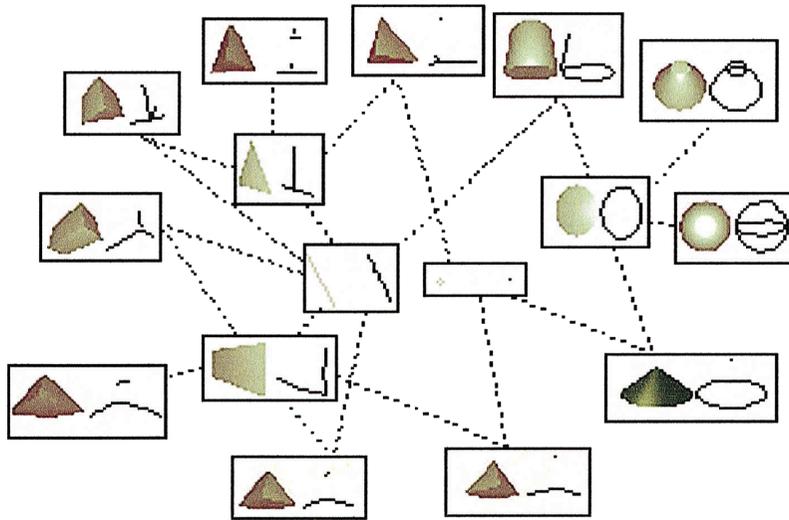


Fig. 5. 3D-sketch language for primitive creation.

following heuristic assumption: If this distance is below some threshold, the stroke can be assumed as being closed, thus a closed freeform-face is reconstructed. If the distance between the two points is above the threshold, it is assumed that the stroke outlines the silhouette of a body of revolution, which can be reconstructed. These generalizations help to keep the grammar small.

The parser stack (lines 1–3 in Table 1) contains the property space of the current stroke, the current property space of the derived object, and the current parse tree. The single stroke properties contain geometric or dynamic gesture information, as discussed in Section 3.3, while the object properties contain object-specific parameters (such as type, position, orientation, size, etc.) that are needed for reconstruction. Additionally, the parse-tree stores history-information (i.e. the hierarchically derived properties of sub-objects).

3.5. Other modalities

Speech is used in our setup as secondary modality, mainly to complement the gestural information. For this, we apply an off-the-shelf software package for continuous speech recognition (IBM's ViaVoice). As for sketch recognition, a grammar has to be defined in advance to outline the recognizable speech language (see Table 2, for example). To avoid confusion with non-speech commands (e.g. caused by conversations with other users that are received by the speech engine), we offer speech input over a regular telephone (the audio data is transmitted to the speech engine via the phone line). This enables us to put the receiver away any time, if we do not want to input speech, and, by taking advantage of a distributed modality processing, it supports a completely remote recognition of verbal commands.

Another solution to this problem is to define keywords at the beginning of the grammar (e.g. 'listen') that explicitly trigger the speech recognition process (see line 1 in Table 2). Although this is not very intuitive, it allows us to use a more ergonomic headset instead of picking up the telephone, every time we want to give a speech command.

With respect to the defined grammar, speech commands can be complete sentences, phrases or single words. The simple example grammar that is shown in Table 2 defines commands for selecting material and texture information from predefined categories and for browsing within a selected category, commands to switch between handwriting recognition and sketch-recognition, and a command for initiating the measuring of created objects. Note, that the *free form* speech command that is passed to the sketch parser (see line 21 in Table 1) has also been defined here (see line 3 in Table 2).

3.6. Modality-merge and context knowledge

This level merges the different information extracted from the single modalities, as well as the provided context knowledge. With respect to the active-tool context, ErgoSketch [6], for instance, applies speech recognition and sketching independently. Therefore, spoken commands or corresponding sketches are used to switch between modes, to select items or to input commands that are usually typed with the keyboard. In addition to this, we merge the verbal and gestural information in a complementary manner. To merge the different modalities, we pass recognized speech information as tokens and information nodes (compatible to the sketch parse tree) into the sketch parser (see line 21 in Table 1 and line 3 in Table 2 for example) and process them as described

Table 1

Simple BNF-grammar to define the sketch language, illustrated in Figs. 5 and 6

1:	parser_stack	= {STROKE_PROPERTIES	stroke_properties
2:		OBJECT_PROPERTIES	object_properties
3:		PARSE_TREE	parse_tree}
4:	< sketched_object >	= < point > < line > < circle > < two_line_face > < three_line_body >	
		< cylinder >	
5:		< two_circle_body > < cone > < two_line_point_body > < four_line_body >	
6:		< single_freeform_stroke_shape > < extruded_freeform_face >	
7:		< freeform_extruded_std_face > < freeform_extruded_freeform_face >	
8:		{return object_type, object_properties and parse_tree}	
9:	< freeform_extruded_freeform_face >	= < single_freeform_stroke_shape > < single_freeform_stroke_shape >	
10:		{return type (<i>freeform extruded freeform face</i>) and compute	
11:		object_properties, assemble parse_tree}	
12:	< freeform_extruded_std_face >	= < two_line_face > < single_freeform_stroke_shape >	
13:		{determine and return object type (<i>freeform extruded triangle or freeform</i>	
14:		<i>extruded rectangle</i>) and compute object_properties, assemble parse_tree}	
15:		< circle > < single_freeform_stroke_shape >	
16:		{return object type (<i>freeform extruded circle</i>) and compute object_properties, assemble	
17:		parse_tree}	
18:	< extruded_freeform_face >	= < single_freeform_stroke_shape > < line >	
19:		{return object type (<i>extruded freeform face</i>) and compute object_properties, assemble	
20:		parse_tree}	
21:	< single_freeform_stroke_shape >	=FREEFORM	
22:		{determine and return object type (<i>freeform face or body of revolution</i>) and compute	
23:		object_properties, assemble parse_tree}	
24:	< four_line_body >	= < two_line_face > < two_line_face >	
25:		{determine and return object type (<i>truncated pyramid or truncated tri-pyramid</i>)	
26:		and compute object_properties, assemble parse_tree}	
27:	< two_line_point_body >	= < two_line_face > < point >	
28:		{determine and return object type (<i>pyramid or tri-pyramid</i>) and compute	
29:		object_properties, assemble parse-free}	
30:	< cone >	= < circle > < point >	
31:		{return object type (<i>cone</i>) and compute object_properties, assemble parse-tree}	
32:	< two_circle_body >	= < circle > < circle >	
33:		{determine and return object type (<i>sphere or truncated cone</i>) and compute	
34:		object_properties, assemble parse-tree}	
35:	< cylinder >	= < circle > < circle >	
36:		{return object type (<i>cylinder</i>) and compute object_properties, assemble parse-tree}	
37:	< three_line_body >	= < two_line_face > < line >	
38:		{determine and return object type (<i>cube, prism or tri-prism</i>) and compute	
39:		object_properties, assemble parse-tree}	
40:	< two_line_face >	= < line > < line >	
41:		{determine and return object type (<i>rectangle or triangle</i>) and compute object_properties,	
42:		assemble parse-tree}	
43:	< circle >	= CIRCLE	
44:		{return object type (<i>circle</i>) and compute properties, assemble parse-tree}	
45:	< line >	= LINE	
46:		{return object type (<i>line</i>) and compute object_properties, assemble parse-tree}	
47:	< point >	=POINT	
48:		{return object type (<i>point</i>) and compute object_properties, assemble parse-tree}	

Table 2
Simple BNF-grammar for a speech language

1:	< STATEMENT > = listen < COMMAND > .
2:	< COMMAND > = new texture < TEXTURE_CATEGORY > new material < MATERIAL_CATEGORY >
3:	texture off free form measure draw next previous < WRITE > .
4:	< WRITE > = write numbers write characters.
5:	< TEXTURE_CATEGORY > = assignable textures miscellaneous stones surfaces
6:	swirls textiles reflect gallery.
7:	< MATERIAL_CATEGORY > = assignable materials autumn rococo sheen glass
8:	metal neon silky spring summer tropical winter.

in Section 3.4. (The only difference here is that the information is forwarded from the speech recognition layer rather than from the dynamic gesture recognition layer.) However, the verbal input does not have to consist necessarily of single words. If it is a complete phrase, we extract the predefined keywords and pass their corresponding tokens to the IMGR sketch parser. This allows us to provide further information to the sketch recognition and interpretation process that, for instance, cannot be sketched or can only be sketched with difficulties (e.g. material information, such as color, texture, etc.).

Sketching freeform strokes is a good example for using both modalities: It is possible to differentiate freeform strokes from the regular elementary strokes within the dynamic gesture recognition layer if their total feature deviation to the best match is above some threshold. However, it is difficult to classify the strokes if they are similar (i.e. if their total feature deviation is below the threshold). To avoid classification conflicts and to support the sketching of any type of freeform stroke, we require an explicit notification in form of a speech command before the freeform stroke is performed. While this verbal token is used for sketch classification, the parameters of the freeform stroke are passed to the gesture parser in form of an information node to support sketch identification.

Fig. 6 illustrates some simple freeform objects that can be created with a mix of gestural input for elementary strokes (e.g., lines, points, circles, arcs, etc.) and verbal input for freeform strokes (see line 21 in Table 1 and line 3 in Table 2). Note that object creation follows the same principles as described in Section 3.4 and is derived from the same sketch language that is illustrated in Fig. 5.

Although our architecture supports a co-verbal gesture interpretation, it does not offer an interpretation of time-stamped verbal and gestural input that occur at the same time, such as described in Bolt's pioneering 'Put-That-There'-Study [23] or other works, such as [24,25]. Even though we think that alternating sequential multimodal information streams are well suited for object creation and interaction within virtual environments, we are interested in extending this level to support parallel multimodal information streams and to evaluate the human factors that are related to this. Predefined or dynam-

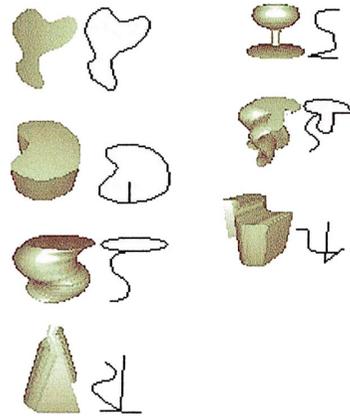


Fig. 6. Simple freeform objects created from representative 3D sketches.

ically growing context knowledge is accessible within this layer and by the sketch parser. It can be used within the context-free grammar that defines the sketch language, as well as after the sketch recognition and interpretation process to derive the intended action.

The context knowledge (implemented as rule-based system) represents application-specific information that has to be known to interact with or to create objects using sketches. It can be generated dynamically (e.g. by sketches themselves, such as the geometric constraints that can be established by sketching them with the Sketch system [2]), or it can be predefined. Examples for partially predefined context knowledge are construction rules for pipes that we modeled as a rule-based system: A piping demonstrator allows pipe components to be connected by sketching single assembly steps. How to connect pipes according to the sketched information (the result from the sketch recognition and interpretation layer), and if they can be connected at all, is determined by inferring the construction rules. A geometric constraint solver finally animates the outlined assembly step and establishes new constraints (i.e. dynamically generated context-knowledge) if possible.

Fig. 7 illustrates a sketched assembly step, the result after inferring the construction rules and solving the

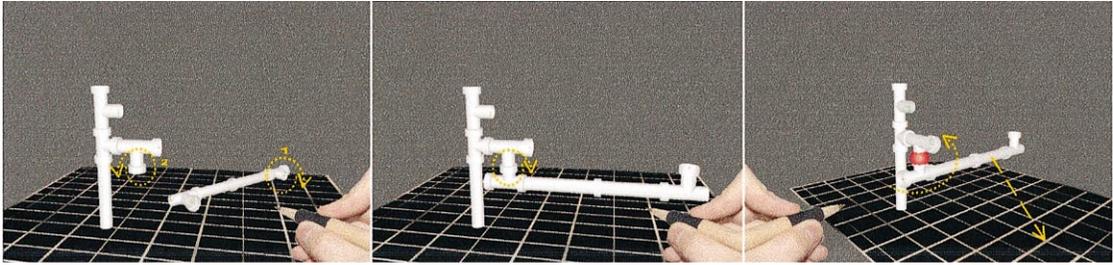


Fig. 7. Sketched assembly step, resulting construction, and constrained interaction.

geometric constraints, and a constrained interaction outlined by sketches.

3.7. High-level interaction

As stated for the low-level interaction layer, the name of this layer refers to its position within our architecture. High-level interaction techniques are sketch-based techniques that are implemented within the scope of the lower layers that are described above.

In the following, we want to introduce some examples for high-level interaction techniques and discuss how they are related to the other layers. *Object Creation* is supported by 2D sketches [13] or by 3D sketches [21]. Sketching on a two-dimensional basis requires a reference plane that can be provided by our transparent pad-like input device [11,12]. The outlined representative sketches are recognized and interpreted before they are used to create standard object primitives or application specific objects. The same principle is applied for unconstrained three-dimensional strokes that are sketched directly within the 3D free-space. The advantage of 3D sketches is that their higher information content (in contrast to 2D sketches, 3D sketches provide depth information) allows us to fully interpret them and completely reconstruct the outlined objects [21]. This is not possible when 2D sketches are used to create 3D objects. Direct sketch manipulation and stroke snapping are the supported low-level interaction possibilities in combination with 3D sketching. However, three-dimensional sketches can also be attached to the local coordinate system of the translucent pad, introducing a three-dimensional sketchpad.

To offer sketch-based *Object Interaction*, 3D sketches can be used to outline basic transformations (e.g. translations, rotations, and scaling). The interpreted sketches reveal information, such as type of transformation, selected object, target position for translation, rotation angle, etc. In addition to basic transformations, context sensitive transformations (i.e. transformations that require context knowledge) can also be sketched, as the assembly example in Section 3.6 shows. Single objects or

groups of objects can be selected by circling them and deleted by scribbling them out, with both 2D and 3D sketches.

System Control is also supported on a two-dimensional or three-dimensional basis. In combination with the sketchpad, 2D sketches can be used to switch between application specific modes, such as context-sensitive menus, coloring mode, window tools, fish net selection with the pad, or object creation. An example for system control with 3D sketches is that the virtual scene can be illuminated by drawing appropriate light sources within the 3D free-space. Type, position, orientation and opening apex can be fully recognized and interpreted from the sketches, while speech information can be used in addition to define color and brightness.

Since speech recognition is still too unreliable to input text without being constrained to a predefined grammar (e.g. object names or filenames, measurements, etc.), we support *Text Input* by recognizing handwriting, similar to what is used with the Virtual Notepad [14]. Handwriting is a strictly two-dimensional task and requires a reference plane, which the translucent pad can offer. Since there is no difference between recognizing strokes that belong to characters or strokes that belong to sketches, we can apply IMGR to manage this. For our system, we decided to train the uni-stroke code of Graffiti [26] (implemented in many of the palm-sized devices, such as Palm-Pilot, WordPad, etc.) to IMGR and to realize the Graffiti finite state machine. We have chosen to do this for two reasons: Graffiti's increasing level of use and the simplicity of the uni-stroke characters. Uni-stroke characters do not require users to define a context-free grammar, and the characters can simply be handled by the dynamic gesture recognition layer. Thus, fully adaptive handwriting recognition can be realized. In contrast to standard Graffiti, we analyze the user's writing behavior during runtime to support an automatic adaptation to it. Thus, starting with a pre-trained version of the character set, an implicit and seamless training is facilitated. Note that multiple users have to activate their individual profiles before they use the system. This can be triggered via speech commands.

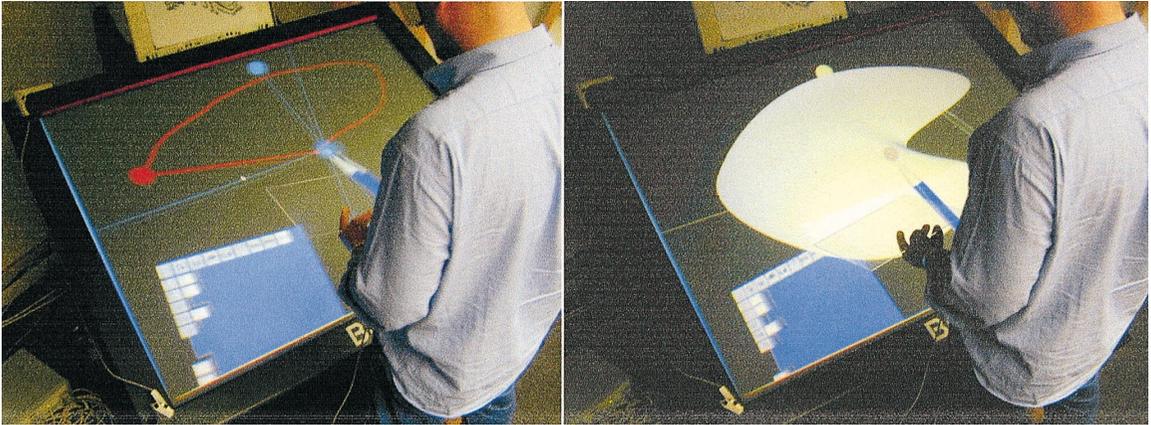


Fig. 8. Freeform sketching: creating a Coons-patch.

Sketching also allows users to generate *freeform surfaces* from hand gestures in free space. Conventionally, freeform surfaces require deep knowledge about their underlying concepts on the user's side (e.g., the user has to know the meaning of control points, knot vectors, u - and v -direction and the impacts of changing any of them). Sketching curves and surfaces into free space provides much more intuitive interface than control-point manipulation. For this reason we explored the possibilities of sketching for freedom surface generation. The aim was to support a seamless creation of well-known freeform surface concepts, such as Coons patches [27], skinned surfaces [28], and net surfaces [29]. Therefore, sketching techniques that do not place any prerequisites on the user's side were developed. The mapping of the input data onto the requirements of different freeform surface types is done by the system in performing some reasoning where necessary (see [30] for details). By using the transparent pad as a virtual mirror, one can generate symmetric surfaces (Fig. 8).

3.8. Applications

The applications that are described in this section offer sketching by embedding our architecture partially or as a whole. Since the applications were developed independently on top of different hard- and software platforms (implementing their existing, non-sketch-based user-interface), they respectively apply their own interaction device layer (making use of different interaction devices, implementing specific device drivers, etc.). However, since the interface to the low-level interaction layer (i.e. a sequence of stroke samples) is given, the sketch-based interface to the application layer can be provided by the architecture (cf. Fig. 1). Even the low-level interaction layers were application specific, the integration processed smoothly, within a few working days. Each of the applications has an individual and domain-specific pur-

pose that is supported by the possibility of using sketches as an additional, human-centered interaction method. Although the applications do not focus on sketching, the latter enlarges the users' interaction capability — especially within immersive virtual environments.

3.8.1. CADesk — using 2D sketching for generation and manipulation of solid geometries

The Virtual Table presents stereoscopic graphics to a user in a workbench-like setting. We have developed a user interface and new interaction techniques for this device based on the transparent props described above — a tracked hand-held pen and a pad [11,12]. These props, particularly the pad, are augmented with 3D graphics from the Virtual Table's display that can serve as a palette for tools and controls, as well as a window-like see-through interface, a plane-shaped and through-the-plane tool, supporting a variety of new interaction techniques. This section describes an extension of this user-interface design space, which uses the described gestural input to create and control solid geometries for CAD and conceptual design [13]. We have anecdotal evidence, that this new interaction paradigm greatly increases the Virtual Table's suitability for design tasks, especially since traditional CAD dialogue can be combined with intuitive rapid sketching of geometry on the pad. Additionally, the resulting events and objects can be associated with scene details below the translucent tablet. For creative CAD applications, informal user studies employing the talk-aloud protocol with students from the Rhode Island School of Design confirm this notion.

The 2D sketches that are used for object creation were developed to be as intuitive as possible, to facilitate easy memorization. In addition, since the user looks through the transparent pad onto the scene that is displayed on the Virtual Table, the representative sketches have been designed to follow the contours of the top-down

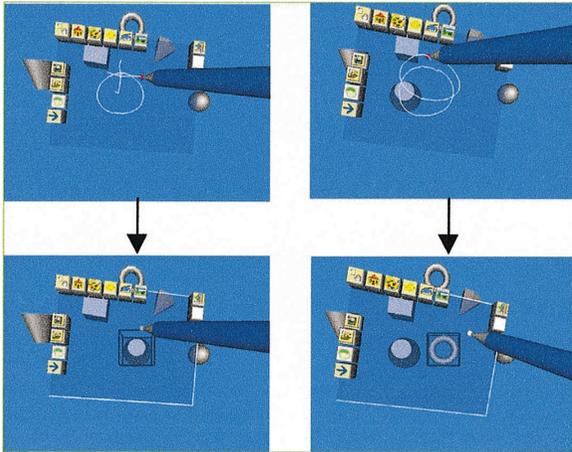


Fig. 9. Creating of solid objects using 2D sketching on a tablet: left, a truncated cone, right, a torus.

projection of the corresponding solid geometries as close as possible.

The currently implemented uni-strokes are conceptually structured in a hierarchical order. A stroke's beginning and end are defined by pressing and releasing a button on transparent pen. Sketches may consist of several pen strokes that are performed close to the pad. These tools are used much like pen and paper, except instead of actually drawing a shape, the computer scans the strokes made on the pad. The strokes' proximity to the pad determines whether or not they contribute to the gesture to be recognized.

The sketches support the generation of 3D objects (cf. Fig. 9) with circular base surfaces, contours (e.g., sphere, cone, truncated cone, cylinder, torus), or rectan-

gular shapes (e.g., cube, pyramid, truncated pyramid). In general, the objects are created by first defining their base surfaces or contours (cf. Fig. 10). Afterwards, a stroke defines either the depth (e.g. for cube) or the height (e.g. for cone). Sketches for truncated solids resemble their non-truncated equivalent in that the height stroke is merely extended by a horizontally cutting finishing-stroke (cf. Fig. 10). Obviously, special solutions must be developed for cylinder, sphere, and torus generation, since these objects would be created using ambiguous sketches.

Our solutions for creating these shapes are defined as follows: the cylinder by two parallel lines that indicate its side view, the torus by two circular strokes, and the sphere by a circular stroke and an arc stroke that indicate the sphere's curvature in all dimensions. Although some sketches show close correspondences, the recognition rate is generally between 95 and 100% (cf. Fig. 10 for the corresponding trained gesture set).

The defined sketches for object manipulation and control are currently limited to the selection and deletion of objects (cf. Fig. 11). Additional control sketches are available that perform mode changes, thus relieving the user interface apparent on the pad from unnecessary dialogue buttons. Fig. 12 shows the different sketches for object control and mode changes. Although several sketches in this group also show close correspondences, the recognition rate is once again between 95 and 100% (cf. Fig. 12 for the corresponding trained gesture set).

To facilitate intuitive interaction and support easy recollection, objects are selected by circling their projected images that are viewed through the pad. (Note: This functionality was actually already supported by the system described in [11], without using the motion-based gesture recognition presented here.) In a similar way, objects are deleted by "crossing them out" on the pad.

		Base shape w/ depth		Base shape w/ height	Truncated basic solid
Rectangular base shape	Gesture				
	Resulting solid				
Circular base shape	Gesture				
	Resulting solid				

Fig. 10. Supported sketches for the creation of solid objects.

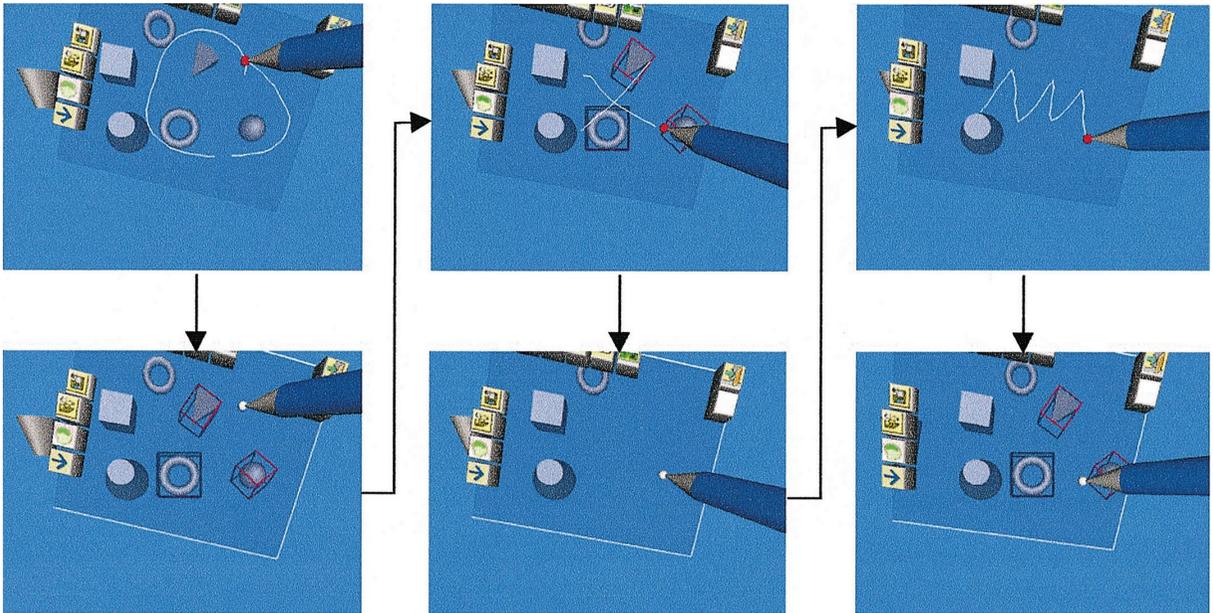


Fig. 11. Sketching for object control. From left to right: object selection, object deletion and undo operation.

Object control	Gesture	X	scribble	circle	diagonal line	
	Resulting operation	delete	undo	select	deselect	
Mode change	Gesture	plus sign	scribble	square	less than sign	caret
	Resulting mode	Context-sensitive menu	Coloring mode	Window tools	Fish net selection	Object creation

Fig. 12. 2D Gestures for object control and mode changes.

Undo is represented by a “scribbling” on the pad, thus resembling the erasure of mistakes on common paper.

3.8.2. Virtual mission planning

This application takes exactly the same approach as the CADesk, in that it uses 2D sketches on a translucent pad to sketch unit symbols of the services (cf. Fig. 13). The difference in this application is that symbols and not certain geometries are retrieved and conceptually organized in a hierarchical sketch language. These symbols are differentiated in visual properties (e.g., texture map or pattern) rather than in geometrical ones. The application allows users inexperienced with VR applications to intu-

itively use the system for strategic planning and Virtual Diplomacy purposes.

This system has so far been demonstrated to divisions of the Navy that are involved in developing new technologies to ease the control of the littoral battlespace as well as Air Force research labs that focus on virtual reality applications to command and control applications. In both cases, the rapidness of interaction and the speed of getting familiar with the system were major criteria that helped in acquiring research projects for the further development of the system. It was often noted that the event of PDA technology and the similarity of the sketching and handwriting recognition provided by

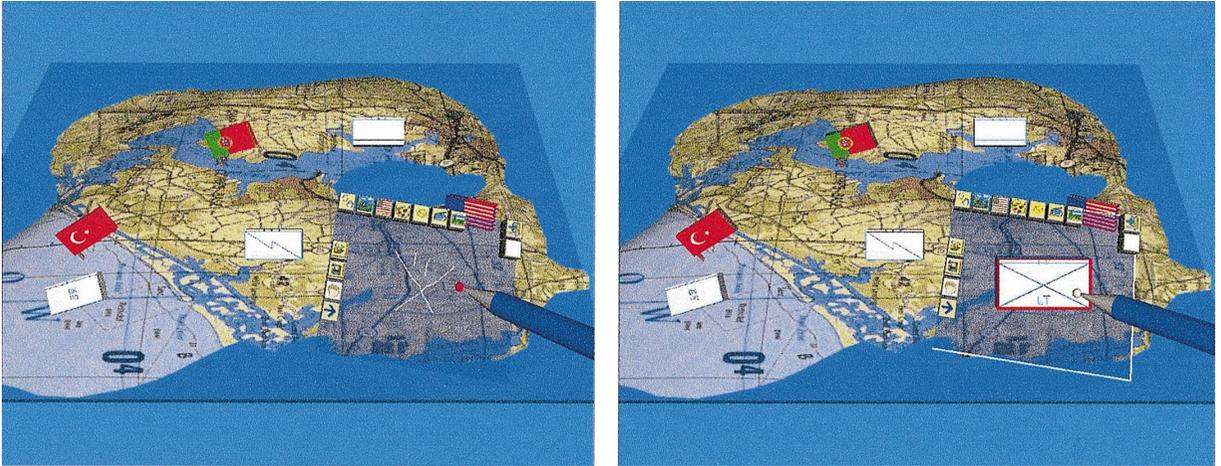


Fig. 13. Virtual mission planning — sketching is used to generate and position service symbols on the fly.

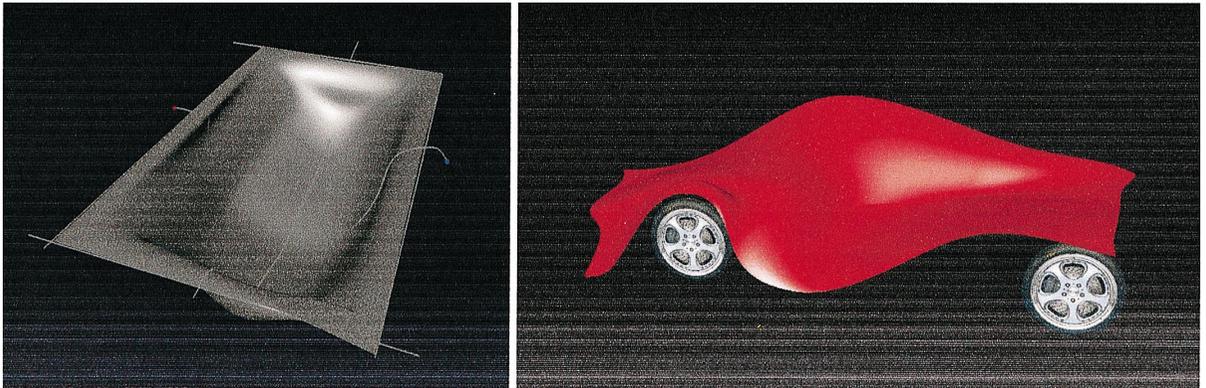


Fig. 14. ARCADE: freeform-sketched bath tub- and car design.

the presented system greatly helped to make the right associations between everyday tools and VR interaction technology.

3.8.3. ARCADE — advanced realism CAD environment

ARCADE is a 3D modeling system and testbed to explore new possibilities in human–computer interaction, such as 3D input, sketching and gesture recognition in the context of 3D modeling tasks. It supports modeling operations, such as 2D/3D primitives, freeform surfaces, sweeping and Boolean operations in combination with 3D input devices (pad and pen) and immersive output (primarily a Virtual Table). In addition to a menu-based preselection, 3D freehand sketching is offered for object creation (primitives and freeform surfaces). While preselection forces the user to alternate between design- and menu space, sketching within the design space allows the user to focus on the design task (Fig. 14).

Although we are just at the beginning of exploring the possibilities of sketching in 3D space, the techniques developed (especially for sketching freeform models) so far cause stunning reactions and interest especially from the car industry, e.g. Opel and Porsche. After demonstrating the system, we made it a habit to hand over the interaction devices to our audience. Even persons, who have never seen or used a CAD system before, are able to sketch objects within minutes. The natural behaviour of the objects, the immediate visual feedback, and the correspondence between hand movements and visual information are reported as the most important advantages over traditional approaches.

4. Conclusion and discussion

In this article, we have described a multi-layered architecture for sketch-based interaction within three-dimen-

sional virtual environments. We have demonstrated that a broad palette of high-level interaction techniques, such as object creation, object interaction, freeform modeling, text input, and environment control can be realized by offering sketching. Rather than developing general sketching applications, these techniques were integrated into existing domain-specific applications to extend their interaction functionality (while maintaining their existing user interfaces) and to evaluate the techniques' applicability.

In contrast to non-immersive desktop approaches, such as Sketch, STILTON, Teddy, and even ErgoSketch (since sketching is supported in a monoscopic mode, only), immersive or semi-immersive three-dimensional environments offer a less constrained sketching (mainly due to the possibility of one- or two-handed 3D interaction, 3D navigation and stereoscopic visual perception). Workbench-like output systems together with pen and pad combinations turned out to be well suited tools, since they support both a constrained 2D, as well as an unconstrained 3D sketching process on an intuitive basis, while representing common and well known everyday items, such as drafting boards, sketch pads, clip boards, and real pens. In the application area of 3D modelling, sketching within free space is a consequent step towards supporting the user in his behavioural and perceptual possibilities. Over the last several thousand years mankind was forced to either physically build any 3D object or to map its shape into a two-dimensional drawing, 3D sketching relieves the users from such encumbers.

In terms of being flexible enough for being integrated with different domain-specific applications, we modularised our architecture — making it possible to solely use required components, to exchange them (while maintaining the interfaces between components — as e.g., illustrated in Figs. 1 and 4), or to distribute them. By not offering a modularised architecture, current sketch applications (such as Sketch, ErgoSketch, STILTON, Teddy, 3-Draw, etc.) are lacking the possibility to exchange components. Being able to exchange single components or sub-components offers an adaptation to a continuously evolving basis technology. This includes, for instance, property and feature descriptions to support different degrees of freedom, classification and training methods to satisfy different recognition demands (e.g. sketch recognition or handwriting recognition), grammars to feature multiple sketch and speech languages, or, on a higher level of abstraction, includes complete architectural layers (such as speech or gesture recognition).

Another major difference to the mentioned related work is that an order-free online or offline adaptive training of single dynamic gestures is supported. New gestures can be trained at runtime, without changing the application's source code. This allows applications or their users to define their own gesture sets in 2D, 3D,

6DOF or any other order (e.g. if gloves are used as input devices), without requiring deep knowledge on gesture recognition. In the case of handwriting recognition, for instance, this features an automatic adaptation to individual writing behaviors of users by the system. Gesture-bases or sketch languages can also be dynamically exchanged at runtime (automatically by the application or interactively by the user) to support task-specific interaction. Other approaches (such as the mentioned related work) that employ hard-coded numerical evaluation of gestures lack in flexibility. In these systems it will be difficult to make the sketch functionality available to a variety of different domains and the associated software applications.

Using grammars as an abstraction for defining speech languages has been proven to be worthwhile in many of the major speech-recognition packages (such as IBM's Via Voice, Microsoft's Speech SDK and others). It turned out that this approach is also efficient for gesture recognition (primarily for sketch recognition). Using tools that generate parsers from predefined domain-specific grammars helps to widely encapsulate the sketch-based interaction metaphors from the application. Not supporting this kind of encapsulation is yet another drawback of hard-coded methods. Furthermore, grammars allow a systematic fusion of multiple modalities and context knowledge — especially, if grammars are already used to define rules for the single modalities (e.g., speech languages and gesture languages).

Sketch-based interaction has not reached the required maturity for many application domains, yet (as it is the case for other new technologies, such as virtual reality). In the future, however, these natural I/O techniques will become more prevalent for use with software applications to solve domain-specific problems. Sketching will mainly be used for virtual reality-aided design (VRAD) tasks, but also to enhance other domains. Together with other input possibilities, such as speech, sketching will become an important part of multimodal interaction. Although we have not carried out formal user studies so far, we received positive feedback from domain experts who are familiar with the interaction possibilities of traditional applications, and who experimented with the introduced applications as well as with the featured sketch-based interaction methods (see Sections 3.8.1–3.8.3). We believe that this can be led back to the fact that a human-centred and natural human-computer interaction (supported by the application of everyday tools and by the execution of habitual everyday tasks) is preferred over the traditional machine-centred human-computer interaction — if (at least) the same results can be achieved.

Overall, by dissociating from WIMP (Windows, Icons, Menus, Pointers) interfaces and moving towards human-centered next-generation user interfaces, sketch-recognition and interpretation will improve, leave the

laboratories and be introduced to the workspaces of architects, designers, analysts, managers, and artists. As it is the case in other engineering disciplines, the development of sketch-based interaction interfaces can — among others — benefit from architectural patterns, such as the ones described in Section 3.

Besides the evaluation of the proposed architecture, a main aspect of our future work will be the design and development of a multimodal agent platform that offers more efficient distribution possibilities and a smooth integration of our architecture into existing and new applications.

References

- [1] Cross N. Natural intelligence in design. Elsevier Design Studies — The International Journal for Design Research in Engineering, Architecture, Products and Systems 1999;20(1):25–39.
- [2] Zeleznik RC, Herndon KP, Hughes JF. Sketch: an interface for sketching 3D scenes. Computer Graphics (Proceedings of SIGGRAPH'96, Annual Conference Series) 1996;30:163–70.
- [3] Igarashi T, Matsuoka S, Tanaka H. Teddy: a sketching interface for 3D freeform design. Computer Graphics (Proceedings, Annual Conference Series, ACM SIGGRAPH) 1999;409–16.
- [4] Turner A, Chapman D, Penn A. Sketching a virtual environment: modeling using line-drawing interpretation. Proceedings of ACM Symposium on Virtual Reality Software and Technology (VRST'99), 1999. p. 155–61.
- [5] Sachs E, Roberts A, Stoops D. 3-draw: a tool for designing 3D shapes. IEEE Computer Graphics and Applications 1991;6:18–26.
- [6] Forsberg AS, LaViola JJ, Zeleznik RC. ErgoDesk: a framework for two- and three-dimensional interaction at the ActiveDesk. Proceedings of the Second International Immersive Projection Technology Workshop, Ames, IA, May 11–12, 1998.
- [7] Bier E, Stone M, Pier K, Buxton W, DeRose T. Tool-glasses and magic lenses: the see-through interface. Proceedings of SIGGRAPH'93, 1993. p. 73–80.
- [8] Schkolne S, Schroeder P. Surface drawing. Technical report CS-TR-99-03, Caltech Department of Computer Science.
- [9] Forsberg A, LaViola J, Markosian L, Zeleznik R. Seamless interaction in virtual reality. IEEE Computer Graphics & Applications 1997;17(6):6–9.
- [10] Barco, Inc., BARON, URL: <http://www.barco.com/project/products/bsp/baron.htm>, 1997.
- [11] Schmalstieg D, Encarnação LM, Szalavári ZS. Using transparent props for interacting with the virtual table. Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics (I3DG'99), 1999. p. 147–153.
- [12] Coquillart S, Wesche G. The virtual palette and the virtual remote control panel: a device and an interaction paradigm for the responsive workbench (TM). In: Rosenbaum L, Astheimer P, Teichmann D, editors. Proceedings of the 1999 IEEE Virtual Reality Conference, March 13–17, Houston, TX, Silver Spring, MD: IEEE Computer Society Press, 1999. p. 213–6.
- [13] Encarnação LM, Bimber O, Schmalstieg D, Chandler SD. A translucent sketch-pad for the virtual table exploring motion-based gesture recognition. Computer and Graphics Forum (Proceedings of EUROGRAPHICS'99) 1999;19(3):277–85.
- [14] Poupyrev I, Tomokuza N, Weghorst S. Virtual notepad: handwriting in immersive VR. Proceedings of IEEE VRAIS'98, 1998. p. 126–32.
- [15] Szalavári ZS, Gervautz M. The personal interaction panel — a two-handed interface for augmented reality. Computer Graphics Forum (Proceedings of EUROGRAPHICS'97) 1997;16(3):335–46.
- [16] Schmalstieg D, Fuhrmann A, Szalavári ZS, Gervautz M. Studierstube — an environment for collaboration in augmented reality. Proceedings of Collaborative Virtual Environments '96, and Virtual Reality Systems — Development and Applications, vol. 3 (1), 1996. p. 37–49.
- [17] Bimber O. Continuous 6D gesture recognition: a fuzzy-logic approach. Proceedings of the Seventh International Conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media (WSCG'99), vol. 1, 1999. p. 24–30.
- [18] Rumelhart D, Zipser D. Feature discovery by competitive learning. Parallel distributed processing. Cambridge, MA: MIT Press, 1986.
- [19] Oja E. A simplified neuron model as principle component analyzer. Journal of Mathematical Biology 1982;15: 267–73.
- [20] Sanger T. Optimal unsupervised learning on a single-layer feed-forward neural network. Neural Networks 1989;2:459–73.
- [21] Bimber O. Rudiments of a 3D freehand sketch based human-computer interface for immersive virtual environments. Proceedings of Virtual Reality Systems and Technology (VRST'99), 1999. p. 182–3.
- [22] Aho AV, Sethi R, Ullman JD. Compilers: principles, techniques, and tools. Reading, MA: Addison-Wesley, 1986, ISBN: 0-201-10194-7.
- [23] Bolt RA. Put-that-there: voice and gesture at the graphics interface. Computer Graphics (Proceedings of SIGGRAPH'80) 1980;14(3):262–70.
- [24] Bolt RA, Herranz E. Two-handed gesture in multi-modal natural dialog. Proceedings of the ACM Symposium on User Interface Software and Technology (USIT'92), 1992. p. 7–14.
- [25] Sparrell CJ, Koons DB. Interpretation of coverbal depictive gestures. AAAI Spring Symposium on Intelligent Multi-Modal Multi-Media Interface Systems, 1994.
- [26] Blinkenstrofer CH. Graffiti. Pen Computing 1995;30–1.
- [27] Coons SA. Surfaces for computer-aided design of space forms. Technical report, MIT, 1967.
- [28] Piegl L, Tiller W. The NURBS book. Berlin: Springer, 1997.
- [29] ACIS geometric modeler application guide. Co: Spatial Technology Inc., 1996.
- [30] Stork A, Schimpke O, de Amicis R. Sketching freeforms in semi-immersive environments. 2000 ASME Design Engineering Technical Conferences & Conference and Informa-

tion in Engineering Conference, DETC 2000, 2000 Sept 10–13; Baltimore, MD. Electronic Proc. on CD-ROM, ASME International 2000.

[31] Viega J, Conway M, Williams G, Pausch R. 3D magic lenses. Proceedings of ACM USIT'96. New York: ACM Press, 1996. p. 51–8.

Mixing Realities in Shared Space: An Augmented Reality Interface for Collaborative Computing

Mark Billinghurst², Ivan Poupyrev¹, Hirokazu Kato³, Richard May²

¹ MIC Research Labs
ATR International
Hikaridai, Seika, Souraku-gun
Kyoto 619-02, Japan
poup@mic.atr.co.jp

² HIT Lab
University of Washington
Box 352142
Seattle, WA, 98195 USA
grof@hitl.washington.edu

³ Faculty of Information Sciences
Hiroshima City University
3-4-1 Ozuka-Higashi, Asaminami-ku
Hiroshima 731-3194, Japan
kato@sys.im.hiroshima-cu.ac.jp

Abstract

In the Shared Space project, we explore, innovate, design and evaluate future computing environments that will radically enhance interaction between human and computers as well as interaction between humans mediated by computers. In particular, we investigate how augmented reality enhanced by physical and spatial 3D user interfaces can be used to develop effective face-to-face collaborative computing environments. How will we interact in such collaborative spaces? How will we interact with each other? What new applications can be developed using this technology? These are the questions that we are trying to answer in research on Shared Space. This paper provides a short overview of Shared Space, its directions, technologies and applications.

Keywords: augmented reality, physical interaction, computer vision tracking, collaboration, entertainment.

1. Introduction

In the Shared Space project, we explore, innovate, design and evaluate future computing environments that will radically enhance interaction between human and computers as well as interaction between humans mediated by computers. In particular, we investigate how augmented reality enhanced by physical and spatial 3D user interfaces can be used to develop effective face-to-face collaborative computing environments.

Shared Space integrates a number of novel interface technologies, including:

Augmented reality. Augmented reality (AR), i.e. overlaying of virtual objects on the real world, allows us to integrate computer-generated and computer-controlled objects into everyday physical reality [6]. Unlike virtual reality where the physical world is completely replaced with synthetic environments, in augmented reality environments, 3D computer graphics objects are mixed with physical objects to become part of the real world.

Collaborative computing. Using computers can be a lonely experience: normally, there is no support for collaborative activities in which several people can work together. In real world collaboration objects and information can be simultaneously and asynchronously accessed by multiple participants, with communication discourse flowing freely between the participants. Shared Space aims to allow for a similar freedom of collaborative interaction that we have in physical environments. We also aim to address some of the limita-

tions of current collaborative interfaces including the lack of spatial cues, the difficulty of interacting with shared 3D data, the introduction of artificial seams into a collaboration, and the need to be physically present at a computer to collaborate [8, 12].

Physical interfaces. Interaction with today's graphical user interfaces (GUIs) is often dubbed as *direct*, meaning that the user "picks" and "manipulates" interface objects using a mouse similarly to how we actually pick and manipulate physical objects. When compared to early command line interfaces, interaction in current GUIs is indeed more direct, nevertheless it can only be loosely compared to our interaction with the physical world. In fact, interface objects do not have physical properties, and "picking" and "manipulating" them are simply metaphors that help us understand how to use the interface by drawing from our everyday experiences. Shared Space investigates the use of *physical, tangible* interfaces [9] where the user can control the computer by physically manipulating multiple simple physical objects that become a part of the user interface.

Spatial 3D user interfaces. 3D user interfaces, an important topic in virtual reality, explore how users can efficiently and effectively interact in spatial 3D computer-generated environments. In spatial interfaces as well as in the physical world, users are not constrained by the 2D metaphor of conventional desktop user interfaces but can interact freely in space. Shared Space is a 3D user interface that provides the user with rich spatial cues and combines spatial and physical interaction for easy control and manipulation of virtual objects.

Computer vision tracking and registration. Computer vision techniques have recently become very popular in user interface research [7] Shared Space makes heavy use of computer vision techniques for tracking and registering virtual objects in the physical world [2].

The rest of this paper is organized as follows. In the next section, we briefly discuss related work, followed by a more detailed discussion of the technologies involved in Shared Space: *augmentation, collaboration, interaction* and their implementation based on computer vision tracking and registration techniques. We then describe a collaborative application that uses these technologies, a game demonstrated at SIGGRAPH 99.

2. Related work

Shared Space has been inspired by a number of previous research projects in augmented reality and ubiqui-

tous computing , computer supported collaborative work (CSCW), 3D user interfaces and virtual reality, and tangible and physical computing [15]. Our research on Shared Space integrates many of these individual components into an effective interface that can support intuitive face to face 3D CSCW.

While the use of spatial cues and three-dimensional object manipulation are common in face to face communication, tools for three-dimensional CSCW are still rare. One approach is to add collaborative capabilities to existing desktop-based three-dimensional packages. However, a two-dimensional (2D) interface for three-dimensional collaboration can have severe limitations, such as users finding it difficult to visualize depth cues or the different viewpoints of their collaborators [10].

Alternative techniques include using large stereo projection screens to project a three-dimensional virtual image into space, such as in the CAVE system [5]. Unfortunately, images can only be rendered from a single user's viewpoint in this setting, so only one person will see true stereo. While this might be satisfactory for some tasks, such as collaborative viewing, effective face to face CSCW using CAVE is impossible.

Multi-user immersive virtual environments provide an extremely natural medium for three dimensional CSCW. Research on the DIVE project [4], GreenSpace [11] and other fully immersive multi-participant virtual environments has shown that collaborative work is indeed intuitive in such surroundings. Participants can seamlessly exchange and communicate gesture, voice and graphical information. However, most current multi-user VR systems are fully immersive, separating the user from the real world: notes, documents, tools and other artifacts of everyday life cannot be easily accessed from immersive virtual environments.

Unlike other methods for three-dimensional CSCW, augmented reality interfaces can overlay graphics and audio onto the real world. This allows for creation of AR interfaces that combine the advantages of virtual environments and possibilities for seamless interaction with real world objects and other collaborators.

Single user AR interfaces have been developed for computer aided instruction [6], medical visualization [1], information displays and other purposes. These applications have shown that AR interfaces can enable a person to interact with the real world in ways never before possible. However, although AR techniques have proven valuable in single user applications, there has been significantly less research on collaborative, multi-user applications. The AR2 Hockey [12] and the Studierstube project [14] are two of the few exceptions.

On the interface side while the physical and tangible interfaces have been extensively explored [9], there have been few efforts at combining them with spatial 3D interfaces. Finally, computer vision techniques have been extensively used to track and register virtual objects in

augmented reality applications. Our approach was inspired by the work of Rekimoto who developed a technique for robust tracking of 2D markers [13].

3. Shared Space

This section discusses key aspects of Shared Space, i.e., augmentation, collaboration, interaction, and implementation based on computer vision tracking and registration techniques.

3.1 Augmentation

Shared Space uses a head-mounted display (HMD) with a lightweight camera mounted in front of the display. The output from the camera is connected to a computer and then to the HMD so that the user sees the real world through the video image. In the physical environment there are a number of marked cards with square fiducial patterns on them and a unique identifying symbol in the middle of the pattern. When the user looks at these cards, computer vision techniques are used to identify the specific marker, calculate head position and orientation relative to the fiducial marks, and display 3D virtual images so that they appear precisely registered with the physical objects (Figure 1). The details of the implementation are briefly described later in the paper, for a full description see [2].

3.2 Collaboration

Shared Space allows users to refer to physical notes, diagrams, books and other real objects while at the same time viewing and interacting with virtual images. More importantly, co-located users can see each other's facial expressions, gestures and body language thus supporting natural face-to-face communication cues. Thus the Shared Space interface allows multiple users in the same location to simultaneously work in both the real and virtual world (Figure 2). Since all users share the same database of virtual objects, they see the same virtual objects attached to the markers from their own viewpoints. Users can pick up and show cards to the other participants, or pass or request virtual objects in the same manner that we do with real objects.

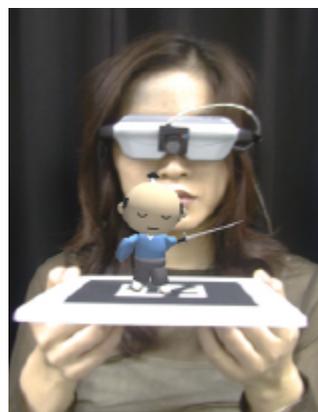


Figure 1: The HMD and a camera are used for registering and viewing virtual objects. Here a samurai model



Figure 2: In collaborative environment of the Shared Space, users can see and interact with physical and virtual objects, and also see the other participants.



Figure 3: Spatial interaction in Shared Space: users trigger animation of virtual objects (in this case the alien and UFO) by bringing two cards together.

3.3 Interaction

Shared Space explores the use of spatial and physical interaction in augmented environments: the user can directly manipulate virtual objects by manipulating marked physical objects with virtual objects on them (Figure 1). The system can robustly track the motion of the physical markers and keep the virtual object precisely aligned relative to the marker. Several markers can be tracked simultaneously so the relative positions of marked objects to each other can be used to trigger virtual object interactions. For example, placing a card with a virtual UFO on it next to one with an alien may trigger an animation of the alien flying in the UFO (Figure 3). There are a wide range of spatial relationships and physical object interactions (shaking, rotating, etc..) that may be used for virtual interactions.

3.4 Tracking and registration

Shared Space uses a computer vision based tracking algorithm designed by Hirokazu Kato [2]. By tracking rectangular markers of known size the relative camera position and orientation can be found in real time. Once this is known, the virtual camera can be placed at the same position so 3D computer graphics objects appear to be exactly attached to markers (Figure 4).

4. Applications and user experiences

A number of applications have been developed and explored using various components and configurations of the Shared Space technology, including a mobile AR conferencing space for remote users [3]. In this section we describe a collaborative entertainment application, which was demonstrated at the Emerging Technologies exhibit at the SIGGRAPH99 conference. The goal of this demonstration was to show how augmented reality could be used to enhance face-to-face collaboration in a way that could be used by novices with no training.

A multi-player game similar to the game “Concentration” was designed. We presented visitors with sixteen 5x7 inch playing cards with tracking patterns on one side, and the visitors were required to match cards. The cards were placed on a table that up to three people could gather around. Each user wore a HMD with a camera attached connected to a computer as previously described. When players turned the cards over they saw a different 3D virtual object on each card, such as a witch, horse, alien, or crabs (Figure 3). The goal of the game was to match objects that logically belonged together, such as an alien and UFO. When cards that matched were placed side by side, an animation was triggered involving the objects on the card. For example, when the card with the virtual witch on it was placed next to the card with a virtual broom on it, the witch would jump on the broom and start to fly around in a circle. Sound cues were also played corresponding to the different animations cued. Since the players were all co-located they could easily see each other, and the virtual objects.

Over the course of the week of August 7-13 around 3000 conference participants tried the exhibit. Users had no difficulty with the AR interface and exhibited the

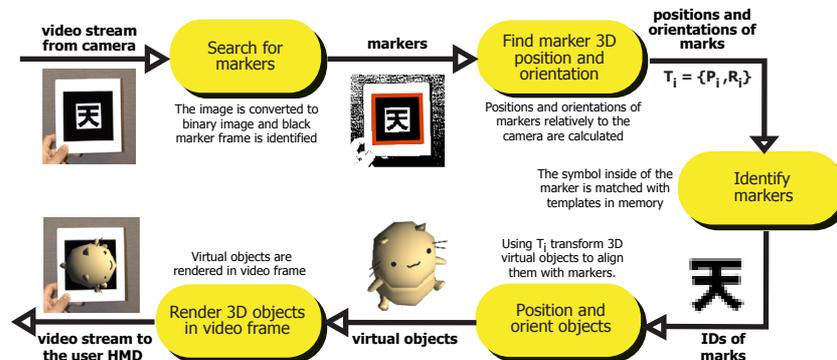


Figure 4: Shared Space tracking and registration technique.

same sort of collaborative behavior seen in typical face-to-face interaction with physical objects. For instance, during play, players would often spontaneously collaborate with strangers who had a matching card, request and pass cards around as well as collaboratively view objects and completed animations. Furthermore, since the matches were not obvious, users would often request and receive help from other collaborators.

The physical, tangible nature of our interface made collaborative interaction very easy and intuitive. Users passed cards between each other, picked up and viewed virtual objects from all angles and almost always expressed surprise and enjoyment when they got a match and the static virtual objects came to life. By combining a tangible, physical interface with 3D virtual imagery, we found that even young children could play and enjoy the game (Figure 5). Users did not need to learn any complicated computer interface or command sets – the only instructions people needed was to turn the cards over, not cover the tracking patterns, and find objects that matched.

Users also commented on how much they liked the image recognition and on how little lag there was in the system. This comment is interesting because there was actually a significant (200-300ms) delay, however, the users became so immersed they did not notice this.



Figure 5: A child in the Shared Space game

5. Conclusions

In our work on Shared Space, we combine real and virtual worlds to create compelling 3D collaborative experiences in which the technology transparently supports normal human behaviors. It is this transparency that is a key characteristic of the Shared Space research and should enable the continued development of innovative collaborative AR interfaces in the future. In the future we plan on investigating how our tangible augmented reality approach can support seamless transitions between physical reality and immersive virtual reality in a collaborative setting. For more information see: <http://www.mic.atr.co.jp/~poup/research/ar/> or http://www.hitl.washington.edu/research/shared_space/

6. Acknowledgments

We are very thankful to ATR computer graphics artists K. Nakao and J. Kurumisawa who designed models

and animations for the SIGGRAPH exhibition. We are also grateful to Shigeo Imura for his help in designing and implementing the computer graphics parts of the software platform as well as all HITL and ATR researchers and management, especially Prof. Furness, Dr. Ohya and Dr. Nakatsu for their support of the project, and HITL students and researchers who prepared and worked the SIGGRAPH demonstration.

7. References

1. Bajura, M., Fuchs, H., Ohbuchi, R., Merging Virtual Objects with the Real World: Seeing Ultrasound Imagery Within the Patient. *Proceedings of SIGGRAPH '92*. 1992. ACM. pp. 203-210.
2. Billinghurst, M., Kato, H., Collaborative Mixed Reality. *Proceedings of ISMR '99*. Springer Verlag. pp. 261-284.
3. Billinghurst, M., Kato, H., Real World Teleconferencing. *Proc. of CHI'99, Extended Abstracts*. ACM. pp. 194-195.
4. Carlson, C., Hagsand, O., DIVE - A Platform for Multi-User Virtual Environments. *Computers and Graphics*, 1993. 17(6): pp. 663-669.
5. Cruz-Neira, C., Sandin, D., Defanti, T., Kentyon, R., Hart, J., The CAVE: Audio Visual Experience Automatic Virtual Environment. *Communications of the ACM*, 1992. 35(6): pp. 65.
6. Feiner, S., MacIntyre, B., Seligmann, D., Knowledge-Based Augmented Reality. *Communications of the ACM*, 1993. 36(7): pp. 53-62.
7. Freeman, W., Anderson, D., Beardsley, P., Dodge, C., Roth, M., et al., Computer vision for interactive computer graphics. *IEEE Computer Graphics & Applications*, 1998. 18(3): pp. 42-53.
8. Ishii, H., Miyake, N., Toward an Open WorkSpace: Computer and Video Fusion Approach of TeamWorkstation. *Comm. of the ACM*, 1991. 34(12): pp. 37-50.
9. Ishii, H., Ullmer, B., Tangible bits towards seamless interfaces between people, bits and atoms. *Proceedings of CHI97*. 1997. ACM. pp. 234-241.
10. Li-Shu, Flowers, W., Teledesign: Groupware User Experiments in Three-Dimensional Computer Aided Design. *Collaborative Computing*, 1994. 1(1): pp. 1-14.
11. Mandeville, J., Davidson, J., Campbell, D., Dahl, A., Schwartz, P., et al., A Shared Virtual Environment for Architectural Design Review. *Proceedings of CVE '96 Workshop*. 1996.
12. Ohshima, T., Sato, K., Yamamoto, H., Tamura, H., AR2Hockey: A case study of collaborative augmented reality. *Proc. of VRAIS'98*. 1998. IEEE. pp. 268-295.
13. Rekimoto, J., Matrix: A Realtime Object Identification and Registration Method for Augmented Reality. *Proceedings of Asia Pacific Computer Human Interaction (APCHI'98)*. 1988.
14. Schmalstieg, D., Fuhrmann, A., Szalavari, Z., Gervautz, M., Studierstube - An Environment for Collaboration in Augmented Reality. *Proc. of CVE '96 Workshop*. 1996.
15. Weiser, M., Some computer science issues in ubiquitous computing. *Comm of ACM*, 1993. 36(7): pp. 75-84.

Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments

Jun Rekimoto

Sony Computer Science Laboratories, Inc.
3-14-13 Higashigotanda, Shinagawa-ku,
Tokyo 141-0022 Japan
Phone: +81 3 5448 4380
Fax: +81 3 5448 4273
E-Mail: rekimoto@acm.org
<http://www.csl.sony.co.jp/person/rekimoto.html>

Masanori Saitoh

Department of Computer Science,
Keio University
3-14-1 Hiyoshi, Kohoku-ku,
Yokohama, Kanagawa 223 Japan
saitoh@aa.cs.keio.ac.jp

ABSTRACT

This paper describes our design and implementation of a computer augmented environment that allows users to smoothly interchange digital information among their portable computers, table and wall displays, and other physical objects. Supported by a camera-based object recognition system, users can easily integrate their portable computers with the pre-installed ones in the environment. Users can use displays projected on tables and walls as a spatially continuous extension of their portable computers. Using an interaction technique called hyperdragging, users can transfer information from one computer to another, by only knowing the physical relationship between them. We also provide a mechanism for attaching digital data to physical objects, such as a videotape or a document folder, to link physical and digital spaces.

KEYWORDS: multiple device user interfaces, table-sized displays, wall-sized displays, portable computers, ubiquitous computing, architectural media, physical space, augmented reality

INTRODUCTION

These days people can take small yet powerful computers anywhere at anytime. Modern notebook-sized portable computers have of several gigabytes of disk storage, processing power almost equal to desktop computers, and an integrated set of interface devices (LCD screen, keyboard, and pointing device). Therefore, it is not impossible to store and carry almost all one's personal data (documents, presentation slides, or digital images) in such a small computer.

In parallel with this tendency, our working environments, such as meeting rooms, are going to be equipped with many computing facilities such as data projectors and digital

whiteboards. It is becoming quite common during a meeting to make a presentation using a video projector to show slide data stored in the presenter's portable computer. It is also very common for meeting attendees to bring their own computers to take notes. In the near future, we also expect that meeting room tables and walls will act as computer displays. Eventually, virtually all the surfaces of the architectural space will function as computer displays [8]. As Lange et al. [5] pointed out, large and multiple display surfaces are essential for supporting collaborative, or even individual, activities. We can simultaneously spread several data items out on these surfaces without hiding each other.

Considering these two trends, the natural consequence would be to support smooth integration between portable/personal and pre-installed/public computers. However, in today's computerized meeting rooms, we are often frustrated by poor supports for information exchange among personal and pre-installed computers. In our physical lives, it is quite easy to circulate physical documents among meeting participants and spread paper diagrams on the table, or hang them on the wall. During a meeting, participants around the table can quickly re-arrange these diagrams. When they are displayed on computer screens, information exchanges between computers often require tedious network settings or re-connection of computers. It is not easy to add annotations to an image on the projector screen while another participant is presenting his data on that screen. When you want to transfer data from your computer to others', you might need to know the network address of the target computer, even if you can physically identify that computer.

In this paper we describe our design and implementation of a computer augmented environment that allows a user to smoothly interchange digital information between their portable computers and a computerized table and wall. Using the combination of camera-based marker recognition and interaction techniques called hyperdragging and anchored cursors, users can easily add their own portable computers to that environment. This intuitive, easy-to-use system is just like dragging icons from on screens to another in a single

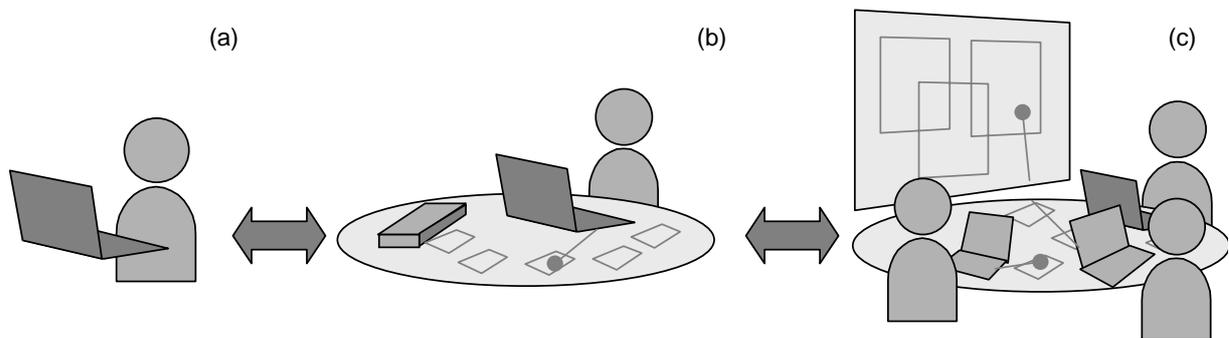


Figure 1: Evolution of spatially continuous workspaces: (a) A user can perform individual tasks with a portable computer. (b) The table becomes an extension of the portable computer. (c) Pre-installed computer displays (table and wall) also serve as shared workspaces for collaborative tasks.

computer supports multiple monitors. People can move information between different computers by only using normal mouse operations and only knowing the physical relationship among them. The system also provides a mechanism for attaching digital data to physical objects, such as a videotape or a document folder, to make tight connections between physical and digital spaces.

A SPATIALLY CONTINUOUS WORKSPACE

While many research systems on augmented physical spaces use pre-installed computers for interaction, we are more interested in how we can smoothly integrate our existing portable computers with the pre-installed ones.

The key features of our system design can be summarized as follows:

Environmental computers as extensions of individual computers

In our design, users can bring their own portable (notebook or palmtop) computers into the environment and put them on the table. Then, the table becomes an extended desktop for the portable computers (Figure 1). That is, the user can transfer digital objects or application windows to the displays on table/wall surfaces. They can use a virtually bigger workspace around the portable computer.

The user manipulates digital objects on the table (or on the wall) using the input devices (such as a track-ball or a keyboard) belonging to the portable computer. Instead of introducing other interaction techniques such as hand-gesture recognition, we prefer to use portable computers because notebook computers already have an integrated set of interaction devices that are enough for most applications. With these interaction devices, users do not have to change user-interface style while dealing with the table or wall. In addition, many recent sub-notebook computers have audio I/O devices, so they can also be used to create voice notes during the task.

If two or more users sit at the same table, the table also becomes a shared workspace among them; the participants can freely interchange information among the participating

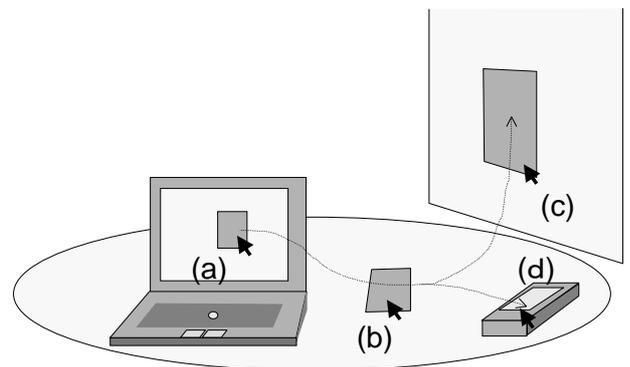


Figure 2: Hyperdragging: A spatially continuous interaction technique for moving information between computers. (a) A user can start moving an object on a computer in the normal manner by dragging it with the pointing device. (b) When the cursor reaches the edge of the screen, it "jumps" to the table surface. (c) The user can continue to drag it to another surface, such as a wall. (d) The user can also drop an item on a physical object, such as a VCR tape, to make a link between real and virtual objects.

portable computers by placing information items on the table/wall.

Support for links between digital information and physical objects

In addition to providing support for portable computers, the system allows users to put non-electronic objects such as VCR tapes or printed documents on the table. By reading an attached visual marker on the object, the system recognizes it and displays digital data that is linked to that object. The user can also add other digital information by simply dragging-and-dropping it onto the object.

Although other systems also support links between physical and digital objects (such as InfoBinder[15], mediaBlocks[18], and Passage[7]), these objects are only for carrying digital data and there are no particular roles in a real world. On the other hand, we are more interested in making a link between digital contents and things *that also have specific roles in the real world*. For example, we can attach editorial instructions



Figure 3: A meeting with InfoTable and InfoWall

to a VCR tape, as a digital voice note. We can also bind physical documents and digital data in a single document folder.

Spatially Continuous Operations

During these operations, we pay special attention to how the physical layout of objects (computers and other real objects) can match the digital manipulations. In other words, the user can use the integrated spatial metaphor for manipulating information in the notebooks, on the table or wall surfaces, and other physical objects placed on the table (Figure 2). For example, when the user wants to transfer data from a notebook computer to the table, he/she can simply drag it from the notebook screen to the table surface across the boundary of the notebooks. At the edge of the notebook screen, the cursor automatically moves from notebook to table. The user can also attach digital data to the physical object by simply dragging and dropping it onto the physical object.

INFOTABLE and INFOWALL: A PROTOTYPE HYBRID ENVIRONMENT

To explore the proposed workspace model, we developed a computer-augmented environment consisting of a table (called InfoTable) and a wall (called InfoWall) that can display digital data through LCD projectors. Figure 3 shows the system configuration of our environment. In this environment, users can dynamically connect their portable computers to perform collaborative and individual tasks. This section summarizes the user-interface features of the system.

We make some assumptions about the portable computers that can be integrated into the environment. To enable the portable computers to be identified by the pre-installed environmental computers, we attach a small visual marker (printed 2D barcode) to each portable computers and other physical object. Portable computers are also equipped with a wireless network for communicating with other computers.

Hyperdragging

When a user sits at the table and puts his/her portable computer on the table, a video camera mounted above the table finds its attached visual marker and identifies the owner of the computer. At the same time, the location of the computer is also recognized.

When the user wishes to show his/her own data to other participants, he/she can use an interaction technique called hyperdragging (Figure 4). That is, the user presses the mouse cursor on a displayed item and drags it toward the edge of the computer screen. When the cursor reaches the edge of the display, it migrates from the portable computer to the table



Figure 4: Moving information using "hyperdragging": A user can drag-and-drop a digital object between a notebook PC and a table surface display. During its operation, an "anchored cursor" line connecting the cursor and the notebook appears on the table display.



Figure 5: The anchored cursor shows the link between information on the table and the notebook computer

surface (Figure 4, middle). If the cursor is grabbing an object, the dragged object also migrates from the portable computer to the table surface. By manipulating the cursor, the user can place the object at any location on the table. Furthermore, the user can move the item toward the edge of the table, to cause a hyperdrag between the InfoTable and the nearby InfoWall display (Figure 4, bottom panel).

This hyperdragging technique supports the metaphor of the table being a spatially continuous extended workspace for portable computers. Users can place data items such as text or graphics around the notebook computer, as if they had a virtually bigger computer desktop.

The combination of two different displays -- a high-resolution small display on the portable computer and a low-resolution large display on the table -- represents the user's focal and peripheral information space. While keeping the focal objects on the notebook screen, the user can spread a number of items around the computer. When the user needs one of them, he/she can immediately hyperdrag it back to the notebook screen.

Anchored cursor

While a user is manipulating his/her cursor outside the notebook computer, a line is projected from the portable computer to the cursor position. This visual feedback is called the *anchored cursor*. When multiple users are simultaneously manipulating objects, there are multiple cursors on the table/wall. This visual feedback makes it easy for all participants to distinguish the owner of the cursors. When two or more participants manipulating objects on the table or on the wall, anchored cursors indicate the owner of the cursor in a visual and spatial way.

The anchored cursor is also used to indicate the semantic relationships between different display surfaces. For example, while the user navigates through a large map projected on the

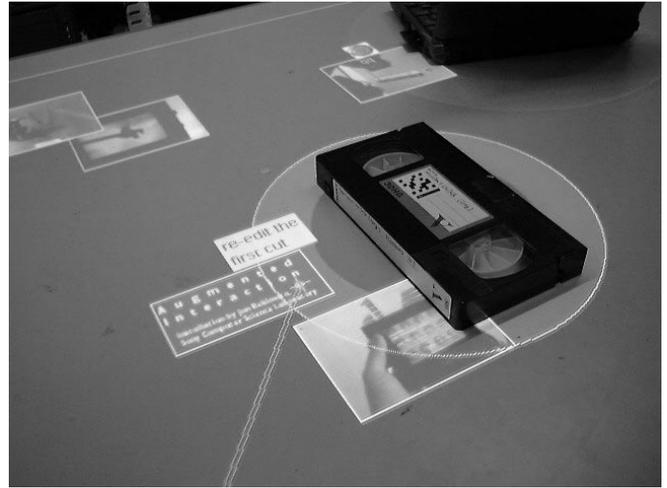


Figure 6: A recognized object (a VCR tape) with an "object aura": A user can attach a digital item by dropping it onto the object aura.

table, a notebook computer continuously displays detailed information related to the current cursor position (Figure 5). The anchored cursor shows the visual connection between them.

Table and wall as shared information surfaces

The InfoTable/InfoWall surfaces can also act as an integrated shared information space among participants. When two or more users sit at the InfoTable, they can freely place data objects on the table from their notebook computers.

Unlike desktop computer's screens, or augmented desk systems [22], there is no absolute notion of the "top" or "bottom" of the screen for table-type computers. Thus the multi-user capability of the InfoTable causes interesting user-interface design issues for determining the above sides. InfoTable uses the recognized spatial position of notebook computers to determine which is the "near" side for each user. For example, when a user brings a diagram from the far side to the near side of the user, the system automatically rotates it so that the user can read it.

Object aura

The system also supports the binding of physical objects and digital data. When an object (such as a VCR tape) with a printed visual marker is placed on the InfoTable, the system recognizes it and an oval-shaped area is displayed at the location of that object. This area, called the "object aura", representing the object's information field (Figure 6). This visual feedback also indicates that the physical object has been correctly recognized by the system.

The object aura represents a data space for the corresponding object. The user can freely attach digital data, by hyperdragging an object from the table surface and dropping it on the object aura. For example, if the user wants to attach a voice memo to the VCR tape, he/she first creates a voice note on his/her notebook computer (using its built-in microphone),

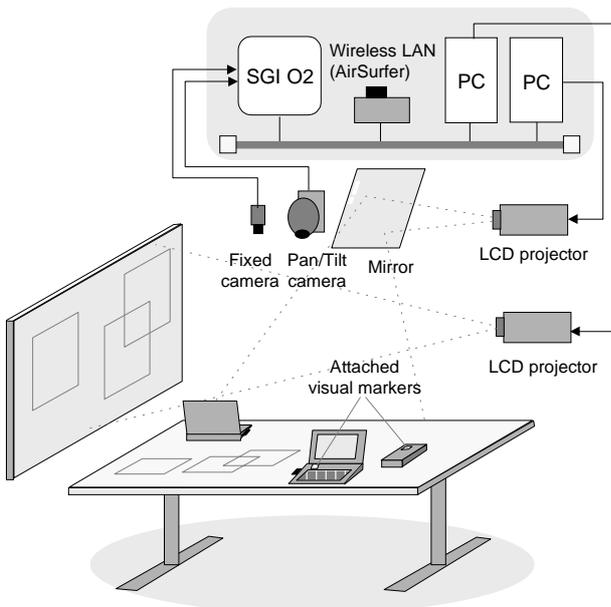


Figure 7: System configuration

and then hyperdrags it from the notebook screen to the VCR tape's aura. When the user releases the mouse button, the voice note is linked to the VCR tape. When someone physically removes the object from the table, the attached data is saved in the network server. This data is re-displayed when the object is placed on any InfoTable.

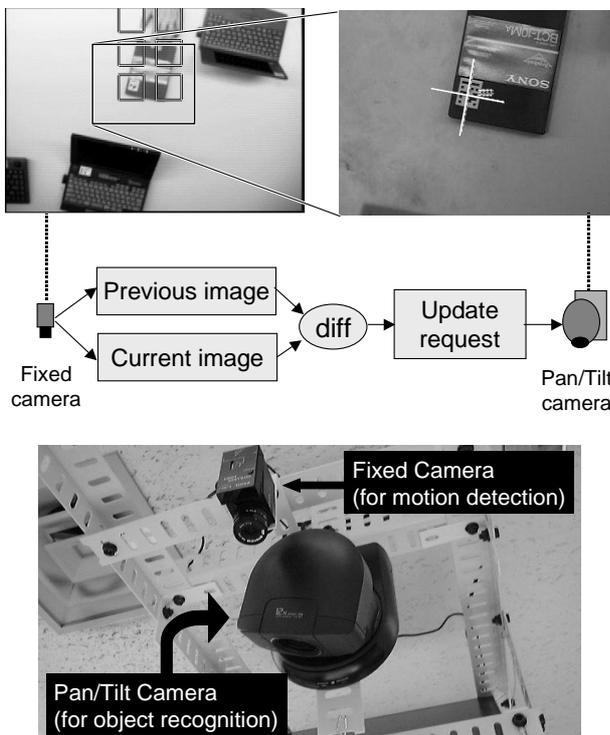


Figure 8: DeskSat uses a combination of two cameras for object recognition

SYSTEM ARCHITECTURE

To enable the interactions described in the previous section, we installed a computer projector and a set of CCD cameras (about 160 cm) above the table. Beside the table, we also installed the combination of a whiteboard and another computer projector as a wall-sized display. Figure 7 shows the device configuration of the system.

Desksat

For the video camera used as an object recognition sensor, there is a tradeoff between camera resolution and the field of view. The camera resolution must be high enough to identify fairly small visual markers that are attached on objects. High-resolution images should also be useful for making a record of the table. However, currently-available video cameras do not cover the entire table surface with the required high resolution. DigitalDesk [22] attempted to solve this problem by adding a second video camera, which is used to capture a fixed sub-part of the desk with higher resolution than the first one. A user is guided to place a document on that focal area.

Our solution is to use a combination of two cameras (Figure 8). The first one is a motor-controlled video camera (Sony EVI-D30) that changes its panning, tilting, and zooming parameters according to commands from the computer. This camera can capture the entire table surface as well as a part of the area with higher resolution (up to 120 dpi) when the camera is zoomed in. Normally, this pan/tilt camera is scanning over the surface of the table by periodically changing the direction and orientation of the camera head. We divided the table surface into a 6-by-6 mesh and the pan/tilt camera is controlled to regularly visit all 36 areas. We called this scheme "Desksat", by analogy to Landsat (land-satellite). In our current setup, it takes about 30 seconds to visit all the areas, including camera control and image processing (marker recognition) times.

The second camera is a fixed camera that is always looking at the entire table surface. This camera analyzes changes on the table from the difference between video images. Then it determines which sub-area has been changed and sends an "area changed" event to the pan/tilt camera. Using this event information, the pan/tilt camera can quickly re-visit the changed area. We choose a threshold value for difference detection so that the fixed camera is not affected by the projected image.

We use a small amount of heuristics to determine the order of visiting these changed areas. Since people normally use the table from the outside, changes in the inner areas are more likely to be object changes. Thus we assign higher priorities to inner areas than to outer areas; when the fixed camera finds several changes simultaneously, the pan/tilt camera checks these areas from inside to outside.

Using these techniques, when a user puts, moves (or removes) objects on the table, this effect will be recognized



Figure 9: Visual marker recognition and obtained position and orientation.

by the system within a few seconds. Although this response time might not be satisfactory for applications that require continuous/realtime object tracking, such as the one in [20], this scheme suits our circumstances quite well where changes occur only intermittently.

Visual marker recognition

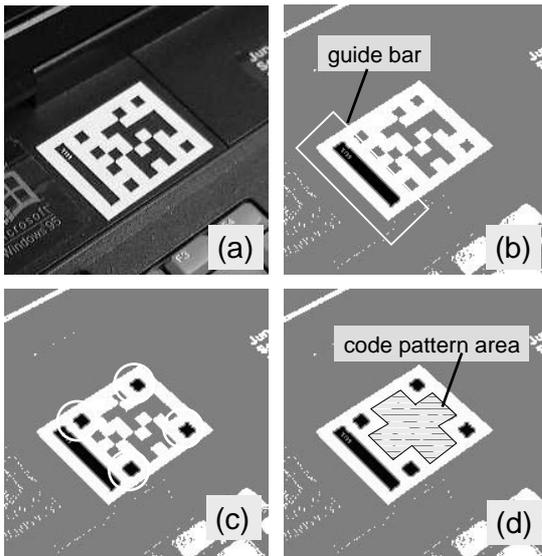


Figure 10: The visual marker recognition algorithm: (a) Original image. (b) Binarized image. Connected regions that have the specific second-order moment are selected. These regions become candidates of a guide bar of the marker. (c) Four corners of the marker region are searched based on the guide bar position/orientation. (d) If the guide bar and the four corners are successfully found, the system finally decodes the bitmap pattern in the marker. Based on the corner positions of the marker, the system estimates and compensates for the distortion effect caused by camera/object tilting. Then the system decodes the code bit pattern. After checking for the error bits, the system determines whether or not the image contains a correct 2D marker.

The printed visual markers (2D matrix code) attached to objects (including portable computers and other non-electronic objects) on the table can identify 2^{24} different objects using the combination of printed matrix patterns (we use a slightly different version of the matrix code system described in [10]). Using the Desksat architecture described above, 2D markers as small as $2\text{cm} \times 2\text{cm}$ can be recognized from the pan/tilt camera above the table.

In addition to its ID being recognized, the marker's position and orientation are also identified (Figure 9). This information is used to calculate object positions in related to the marker position. For example, the position of the cursor on the table while the user is doing a hyperdrag, is calculated based on the current position/orientation of the marker attached on the portable computer. The marker recognition algorithm is summarized in Figure 10.

Since 2D codes cost virtually nothing and can be printed, there are some uses that could not be achieved by other ID systems. For example, we can use small Post-it notes with a 2D code. This (physical) Post-it can convey digital data such as voice notes or photographs with an attached ID.

Hyperdragging

To enable hyperdragging (when the user moves the cursor of the notebook computer from notebook to the table), the system designates mouse-sensitive areas along all four edges of the notebook screen. When the cursor enters this area, the system re-maps the cursor position to the screen, and calculates the offset of this remapping to maintain the cursor position on the table. While the real (original) cursor stays near the edge of the notebook screen, the user can control the virtual cursor position on the table by continuing to press the pointing device.

To correctly calculate the cursor position on the table, the system also has to know the notebook's position and orientation on the table. The system gets this information from an attached visual marker on the notebook PC. Figure 9 shows how the system finds the PC position/orientation based on the attached marker.

Object migration

As a result of hyperdragging, the system needs to transfer data between two computers (e.g., from a notebook computer to the computer running the table display). All application programs for our environment are written in Java and the system employs Java's object serialization mechanism and the remote method invocation (RMI) method to transfer objects. Currently we support text, sound (voice notes), URLs, file short-cuts, and image data as migratable object classes.

EXPERIENCE AND DISCUSSIONS

Up to the time this paper was written, no formal evaluation had been conducted. However, with this environment, the authors and their colleagues in the laboratory have experimentally tried several collaborative activities including a

group meeting.

The concept of hyperdragging was instantly understood by the users and well accepted. Many users were surprised that they could freely move objects between different computers and other physical objects, with a simple drag-and-drop operation. People also appreciated being able to attach data onto the wall surface while sitting at the table. Many wished that they could also move physical objects with the cursor! Anchored cursors were also helpful when two or more users were performing operation simultaneously, especially when the users manipulated object far from their positions. Some users suggested (and we are considering implementing) putting small peripheral devices, such as printers or scanners, on the table and supporting hyperdragging to them. For example, the user could drop an image object onto the printer for making a hardcopy of it.

Some users felt that moving an object across a larger distance was tiresome. We might be able to incorporate techniques other than dragging, such as described in[2]. We also felt that the mapping scale between pointer movement and the pointing device greatly affects usability. Since the projector resolution on the table (about 20 dpi) is much coarser than the notebook computer's (100-110 dpi), mapping without scaling causes a discontinuous change in cursor speed at the boundary between the notebook and the table.

We also observed that there were interesting differences between hyperdragging and our previous multi-device interaction technique called "pick-and-drop"[9, 11]. Pick-and-drop uses a digitizer stylus to pick up a displayed object from one screen and drop it on another screen. Pick-and-drop is a more direct and physical metaphor than hyperdragging, because its operation is quite similar to picking up a real object. Hyperdragging allows a user to manipulate objects that are out of the user's physical reach, while pick-and-drop does not. Pick-and-drop requires a stylus-sensitive surface for operation, but hyperdragging works on any display and projected surfaces.

There is also the question of suitability between pointing devices and interaction styles. Apparently pick-and-drop is best suited for a pen, while hyperdragging does not work well with a pen because it forces indirect mapping between the pen position and the cursor position. On the other hand, hyperdragging is more suitable for a track-ball or a track-point, and these are common for notebook-sized computers.

RELATED WORK

Research on augmenting face-to-face interactions often assumes pre-installed computer facilities so the configuration of computers is fixed. For example, Colab[17] provides a projector screen and table-mounted computers for participants. There was no support for incorporating other computers that the participants might bring to that environment. However, considering recent trends in mobile computing, it would be more practical to support dynamic connections between

mobile and pre-installed computers.

There are several systems that project digital information onto the surface of a physical desk. VIDEODESK[4] consists of a light table and a video camera. The user can interact with the other participant's silhouette projected onto the table. DigitalDesk [21, 22] allows interactions between printed documents and digital information projected on a desk. A recent version of the DigitalDesk series also added a document identification capability based on OCR[13]. Luminous Room[19] (and its underlying "I/O bulb" concept) uses a video projector mounted on a computer-controlled gimbal to change the projection area. Its application called Illuminating Lights[19] helps a holography designer to rapidly layout physical optics devices on the desk. Streitz et al. developed a set of computer augmented elements including a wall, chairs, and a table[7]. Among them, the InteracTable is a table-sized computer supporting discussion by people around it. It also displays information which is carried by a physical block called "Passage". While these systems mainly focus on interaction between non-electronic objects and projected digital information, our system also supports information interchange among portable computers, table/wall surfaces, and physical objects.

The Desksat architecture was partially inspired by the whiteboard scanning system called ZombieBoard[14]. Zombieboard controls a pan/tile camera to capture the mosaic of partial whiteboard images. By joining these images together, a higher resolution image of the entire whiteboard can be produced. The Brightboard [16] is another example of a camera augmented whiteboard system; it recognizes hand-drawn commands made by a marking pen.

As for multi-computer interactions, the Hybrid User Interfaces [1] is an application for a see-through head-mounted display that produces a virtually bigger screen around the screen of the desktop computers. The PDA-ITV system[12] uses a palmtop computer (Apple Newton) as a commander for an interactive TV system. These systems assume a fixed-devices configuration, and are mainly designed for single-user applications.

Ariel [6] and transBOARD[3] support connections between barcode-printed documents or cards and digital contents. Insight Lab[5] is a computer supported meeting room that extensively uses barcoded tags as physical/digital links and commands. These systems normally require a manual "scan" of each printed barcode. This may become a burden for users, especially when they have to deal with a number of barcodes. These systems do not recognize the location of each object, so they require other mechanism to achieve spatially continuous operations.

CONCLUSIONS AND FUTURE WORK

We have described our design and implementation of a hybrid work space, where people can freely display, move, or attach digital data among their computers, tables, and walls.

There are a number of features that must be improved. Currently, we only support Java-based applications and users cannot directly interchange information between other applications that are not written in Java (such as PowerPoint) or native desktop environments (such as the Windows desktop).

We are also interested in implementing a smaller version of InfoTable for individual users. In this environment, user can hyperdrag items from their computer to the wall (typically a cubicle partition) in front of them, in the same way that they usually attach a post-it note to it. When the user wants to attach a To-Do item on the schedule, he/she can simply hyperdrag it to the physical calendar on the wall.

ACKNOWLEDGMENTS

We thank Takahashi Totsuka for helpful discussions and we are also indebted to Mario Tokoro for their continuing support of our research.

REFERENCES

1. Steven Feiner and A. Shamash. Hybrid user interfaces: Breeding virtually bigger interfaces for physically smaller computers. In *Proceedings of UIST'91, ACM Symposium on User Interface Software and Technology*, pp. 9-17, November 1991.
2. Jorg Geisler. Shuffle, throw or take it! working efficiently with an interactive wall. In *CHI'98 summary*, February 1998.
3. Hiroshi Ishii and Brygg Ullmer. Tangible Bits: Towards seamless interfaces between people, bits and atoms. In *CHI'97 Proceedings*, pp. 234-241, 1997.
4. Myron W. Krueger. *Artificial Reality II*. Addison-Wesley, 1990.
5. Beth M. Lange, Mark A. Jones, and James L. Meyers. Insight Lab: An immersive team environment linking paper, displays, and data. In *CHI'98 Proceedings*, pp. 550-557, 1998.
6. W.E. Mackay, D.S. Pagani, L. Faber, B. Inwood, P. Louniainen, L. Brenta, and V. Pouzol. Ariel: augmenting paper engineering drawings. In *CHI'95 Conference Companion*, pp. 420-422, 1995.
7. Torsten Holmer Norbert A. Streitz, Jorg Geisler. Roomware for cooperative buildings: Integrated design of architectural spaces and information spaces. In Norbert A. Streitz and Shin'ichi Konomi, editors, *Cooperative Buildings - Integrating Information, Organization, and Architecture*, 1998.
8. Ramesh Raskar, Greg Welch, Matt Cutts, Adam Lake, Lev Stesin, and Henry Fuchs. The office of the future: A unified approach to image-based modeling and spatially immersive displays. In *SIGGRAPH'98 Proceedings*, pp. 179-188, 1998.
9. Jun Rekimoto. Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments. In *Proceedings of UIST'97*, pp. 31-39, October 1997.
10. Jun Rekimoto. Matrix: A realtime object identification and registration method for augmented reality. In *Proc. of Asia Pacific Computer Human Interaction (APCHI '98)*, July 1998.
11. Jun Rekimoto. A multiple-device approach for supporting whiteboard-based interactions. In *Proceedings of CHI'98*, February 1998.
12. Stott Robertson, Cathleen Wharton, Catherine Ashworth, and Marita Franzke. Dual device user interface design: PDAs and interactive television. In *CHI'96 Proceedings*, pp. 79-86, 1996.
13. Peter Robinson, Dan Sheppard, Richard Watts, Robert Harding, and Steve Lay. Animated Paper Documents. In *7th International Conference on Human-Computer Interaction, HCI'97*, 1997.
14. Eric Saund. ZombieBoard project description. <http://www.parc.xerox.com/spl/members/saund/zombieboard-public.html>.
15. Itiro Siio. InfoBinder: a pointing device for a virtual desktop system. In *6th International Conference on Human-Computer Interaction (HCI International '95)*, pp. 261-264, July 1995.
16. Questin Stafford-Fraser and Peter Robinson. Brightboard: A video-augmented environment. In *CHI'96 proceedings*, pp. 134-141, 1996.
17. M. Stefik, G. Foster, D. Bobrow, K. Khan, S. Lanning, and L. Suchman. Beyond the chalkboard: computer support for collaboration and problem solving in meetings. *Communication of the ACM*, Vol. 30, No. 1, pp. 32-47, 1987.
18. Brygg Ullmer, Hiroshi Ishii, and Dylan Glas. media-Blocks: Physical containers, transports, and controls for online media. In *SIGGRAPH'98 Proceedings*, pp. 379-386, 1998.
19. John Underkoffler. A view from the Luminous Room. *Personal Technologies*, Vol. 1, No. 2., June 1997.
20. John Underkoffler and Hiroshi Ishii. Illuminating Light: An optical design tool with a luminous-tangible interface. In *CHI'98 Proceedings*, pp. 542-549, 1998.
21. Pierre Wellner. The DigitalDesk calculator: Tangible manipulation on a desk top display. In *Proceedings of UIST'91, ACM Symposium on User Interface Software and Technology*, pp. 27-34, November 1991.
22. Pierre Wellner. Interacting with paper on the DigitalDesk. *Communication of the ACM*, Vol. 36, No. 7, pp. 87-96, August 1993.

InfoStick: an interaction device for Inter-Appliance Computing

Naohiko Kohtake¹, Jun Rekimoto², and Yuichiro Anzai¹

¹ Department of Computer Science, Keio University,
3-14-1 Hiyoshi, Kohoku-ku, Yokohama 223-0061 Japan
kohtake@naohiko.com, anzai@aa.cs.keio.ac.jp
<http://www.naohiko.com/>

² Interaction Laboratory, Sony Computer Science Laboratories,
3-14-3 Higashigotanda, Shinagawa-ku, Tokyo 141-0022 Japan
rekimoto@csl.sony.co.jp
<http://www.csl.sony.co.jp/person/rekimoto.html/>

Abstract. Many electric appliances have recently become network reachable, and we would receive better services from them if we could use them in combination. We have therefore developed a new hand-held interaction device called “InfoStick” that serves as an “information carrier” for these appliances. For example, a user can “pick up” TV program information from a web browser and “drop” it into a VCR deck, just like moving a physical object from one place to another. Using attached visual markers, the InfoStick identifies information appliances or other physical objects and gives an appropriate choice of action to the user. This paper explains the design and implementation of the InfoStick as well as several potential applications using this device.

1 Introduction

The network infrastructure has spread all over the world, and nowadays there are a variety of devices that can access the Internet. Internet access is no longer limited to personal computers or powerful workstations. Thanks to recent advances in digital and network technologies, many consumer electric devices such as VCRs, electric-organs or air-conditioners, as well as office appliances such as printers and LCD projectors are becoming “network reachable”. We call these devices “Information Appliances”. It is now reasonable to expect these appliances to communicate with each other in order to provide better services to users. For example, a VCR could receive a TV program information from the web browser or a printer could create a hardcopy of an image projected on a LCD projector.

However, operating these multiple devices may cause user interface problems. We might have to handle a number of remote controllers for each device, and there are few practical ways of controlling two or more appliances. When we want to “transfer” TV program information from the web browser on a computer to

the VCR deck, the VCR controller does not help us. We thus need to operate information appliances in combination, that is not on their own appliance.

In addition, we also need to deal with physical (non-electric) objects such as printed paper. Even if digital and network technology becomes more advantageous, paper still has significant advantages: it is portable, writable, inexpensive, and physically visible. However, transferring data between information appliances and physical objects always requires manual operations. For instance, when we find an URL of an interesting web site on a poster, it is necessary to input the URL into a computer with a keyboard or write it down if there is no computer nearby. It would be quite useful if we could “pick up” a printed URL and “drop” it on an information appliance. In summary, we always need support for exchanging information between digital and physical objects. We call such operations “Inter-Appliance Computing”.

To provide this support, we have developed a hand-held device called “InfoStick” that serves as “an information carrier” for Inter-Appliance Computing. Using the InfoStick, a user can “pick up” TV program information from a web browser and “drop” it into a VCR deck. The InfoStick identifies information appliances or other physical objects by recognizing attached visual markers and gives an appropriate choice of actions to the user.

2 InfoStick Device



Fig. 1. External appearance of the InfoStick

The InfoStick prototype is a hand-held interaction device for exchanging information among information appliances and physical objects. Figure 1 shows

the external appearance of the InfoStick. It consists of a small display to show what kind of data items can be exchanged, a video camera for object recognition, three buttons to operate data exchanges, and a micro processor for controlling all of them. The InfoStick can be connected to the Internet through a wireless network. Physically, it looks like a laser-pointer or a small wand, and can be easily pointed to target objects. Using the InfoStick is similar to drag-and-drop, a commonly used technique for interacting with GUIs.

The InfoStick automatically identifies the information appliance (e.g., a VCR) or the physical object (e.g., paper) in front of it, and shows a user a list of appropriate actions on the display. Visual markers attached to the objects are recognized by the InfoStick's camera. Among the three buttons for operating the InfoStick, the “get” button is used to “pick up” information from the target object, the “put” button is used to transfer information from the InfoStick to the target object, and the “select” button is used to select actions and information shown on the InfoStick display.

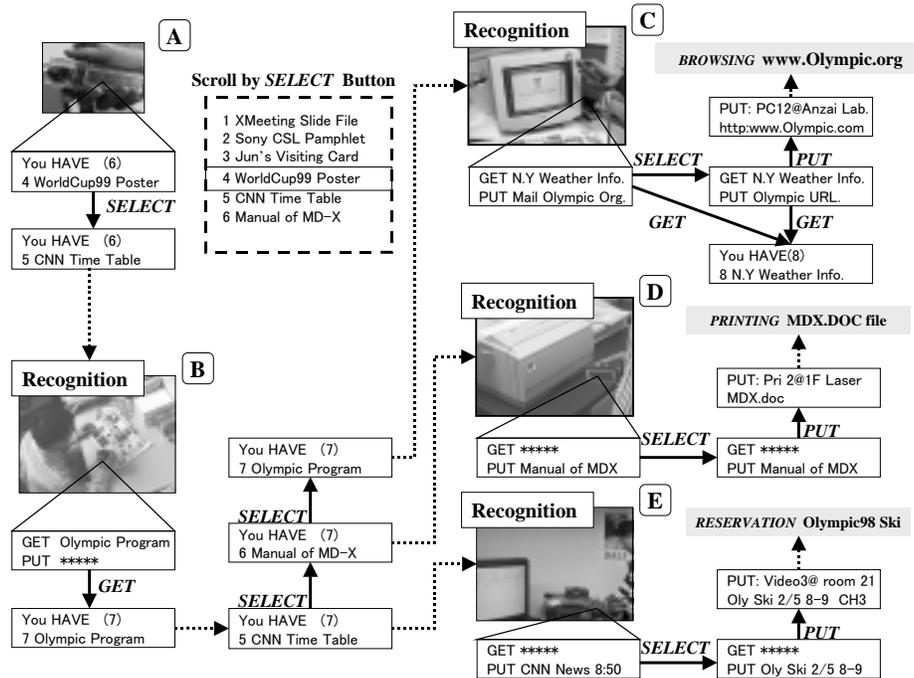


Fig. 2. Typical Information flow during the InfoStick operation

Figure 2 shows a typical information flow during the InfoStick operation. The InfoStick display shows a list of data items, and a user can scroll down the list and select one by pushing the “select” button (Figure 2 A). When a user points the InfoStick at a target object, the video camera mounted in the InfoStick detects a visual marker attached to the object (Figure 2 B). Then, the display on the InfoStick shows a list of items that can be picked up from the target object and also a list of items that can be transmitted to the target object. When the “get” button is pushed, the target object’s information is stored in the InfoStick. In Figure. 2 B, a user is getting “Olympic Program information”.

When a user moves to another target, the InfoStick recognizes it and the available actions corresponding to the recognized object appear on the display (Figure 2 C, D, and E). If the InfoStick has some data that can be “put” into the computer, the user can select an action by the “select” button (from the mail address to the URL in Figure 2 C). By pushing the “put” button, the user can see on the display which target and what kind of information the InfoStick has put into the computer. In this case, the target is a PC “PC12@Anzai Lab.” and the “put” information is “Olympic URL”, which the user can then browse. However, if the InfoStick has only one possible action corresponding to the recognized object, the displayed action does not change even if the user pushes the “select” button (Figure 2 D). So in this case, the user can do only one operation; that is, printing out “Manual of MDX file”. If a user wants to select information previously put in, he can select it before detecting a target object. Then, this information appears on the display as the possible action that he can “put” first (Figure 2 E). After that, other possible actions can be selected in turn.

During these operation sequences, the InfoStick does not directly “get”/“put” data from/to the target objects. Instead, actual data transfer occurs through the network to which all devices are connected. The InfoStick recognizes a target object according to the attached ID and issues appropriate data transfer commands to the network. In our system, we use printed 2D matrix codes (see Figure 1) as IDs. It is also possible to attach such IDs to non-electric objects.

3 InfoStick Applications

Using the prototype InfoStick, we have built several potential applications to accomplish an interaction for Inter-Appliance Computing. We believe this technology will become a part of functions for mobile phone and each user has his own InfoStick device in the future. Some examples of these applications are given in the following sections.

3.1 Transferring Information between Computers

A basic usage of the InfoStick is to transfer digital data between several computers. At a meeting, for example, the presenter often uses a projector to support his presentation (Figure 3).



Fig. 3. At a meeting, the InfoStick creates the illusion that the presenter can “get a slide from the projector” and “put it in my computer”.

The presenter can get the target presentation slide by physically pointing the InfoStick to the projector and pushing the “get” button. When he wants to take the entire slides, he can “select” the “Entire Slides” menu item and push the “get” button. After that, if he wants to create a copy of the acquired slide, he can point the InfoStick to his computer and upload the slide by pushing the “put” button. In this case, the projector itself does not have to hold the slide data. In fact, the actual slide contents are stored in servers on the network, and the projector is used as a physical landmark for obtaining data, because it has a mental connection to the currently displayed slide. The visibility and tangibility of the user’s action are important because of intuitive. A user can therefore exchange information more directly and the InfoStick creates the illusion that a user can “get a slide from the projector” and “put it in my computer”. On the other hand, if the usual file transfer method is used, a user must recognize both the target computer’s name and the slide’s name in order to move the slide by file transfer protocol (FTP). These operations are quite symbolic and thus invisible.

3.2 Operating Information Appliances

The second possible application of the InfoStick allows a user to operate information appliances. For example, if TV program information is stored in the InfoStick, when a user points the InfoStick to the target VCR deck, the display of the InfoStick shows the TV program names that he can reserve for recording. After selecting the program name with the “select” button and putting it into the VCR with the “put” button, the VCR is thus programmed. (Figure 4).

Another InfoStick application is making a phone call. You normally know the person’s name and phone number before calling. If you do not, you have to find them out. Without the phone number, it is impossible to call. However, with the InfoStick, a user does not have to know the number because it is stored



Fig. 4. The InfoStick programs a VCR to make a recording

in a server on the network. The user simply points the InfoStick at a phone and puts the person's name into the phone, which then calls the person.

3.3 Getting information from paper

The InfoStick can also “get” information from physical objects like paper by using visual markers. When the InfoStick recognizes the IDs, a server connected to the InfoStick displays a list of items of the recognized physical object. Then, the user can “get” the information from the target object directly by the same interaction as that with information appliances (Figure 5).



Fig. 5. A user can “pick up” a printed URL from a poster and “drop” it into a computer

3.4 Putting Information onto paper

Another InfoStick application is to attach digital information onto paper like a tag or a document note. For example, it would be useful if we could attach presentation slide files to the corresponding document. Using the InfoStick, the user can “pick up” the slide data from the computer and attach it to the printed marker on the document. For instance, a user can simply take this document to the conference, “get” the presentation file from this marker, and “put” the file into the projector. So the user does not have to bring a computer, a projector or a floppy disk. InfoStick can be used to place all the necessary information on the presentation paper. The InfoStick can also use a piece of paper as a physical memory bank. If we want to store data for a long time, we can “put” it and write down its name as a title on a tag. Later, when a user wants to use this data, he can “get” it from the tag. This system can also be used by teachers for announcing information about exercises to students. After teachers “put” data with the title of the exercise on a tag, students use the InfoStick to “get” it from the tag on a notice board.

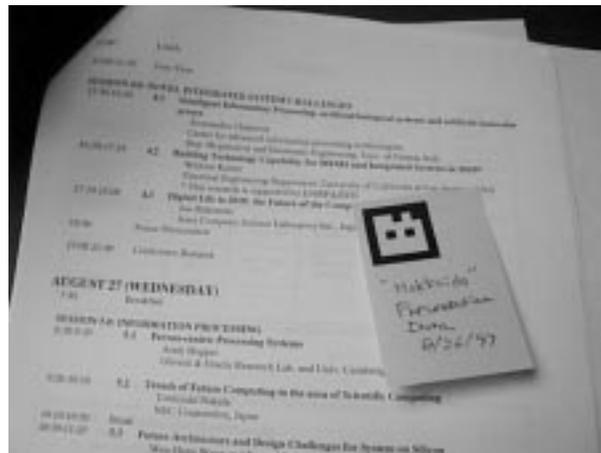


Fig. 6. A PostIt note with attached digital data

It is also possible to use a PostIt note with a printed visual marker (Figure 6). The user can attach any digital data on a PostIt note, and stick it to any objects. This usage provides a way of organizing digital data with physical documents. So even networking becomes ubiquitous, paper will still remain significant. This application of the InfoStick therefore augments electric features with advantages of paper.

3.5 Getting information from one object and Putting it into many objects

In everyday life, there are many kinds of information. A name card is a typical example of this. Generally, it gives a person's name, occupation, address, phone/facsimile number, email and URL. When we make a phone call, we intuitively select appropriate information (a phone number) from the card. When we use a facsimile, the other attribute (a facsimile number) would be used instead of a phone number. There is thus an implicit correspondence between the target device and these attributes. When the InfoStick identifies the target object, it automatically selects appropriate information by combining the attribute's ID and the target object's ID (Figure 7).

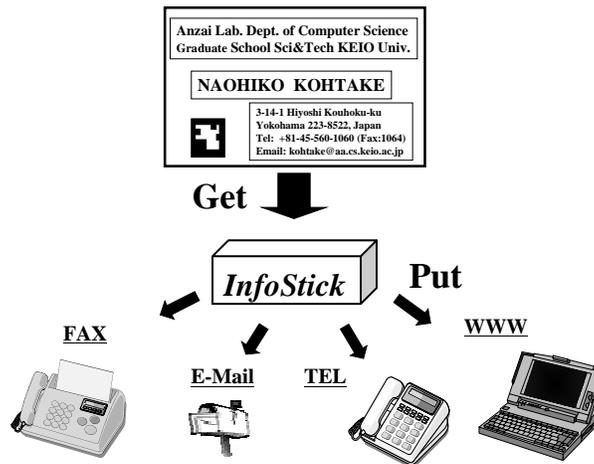


Fig. 7. Automatic selections of information according to the target object

3.6 Getting information from a screen

The InfoStick can recognize visual markers displayed on a screen. It is therefore possible to "pick up" the IDs from TV programs or web pages. Figure 8 illustrates this technique. When a user is watching a TV program, he finds an interesting piece of information on the TV screen. Then, he "gets" this information from the visual marker on the TV screen (above-left) and "puts" it to the nearby computer (below-left). The corresponding web page appears on the computer screen. After browsing the information from this page, he also find some information on an interesting TV program and he "gets" it by pointing the InfoStick at the browser on the screen (below-right). Finally, he goes to the VCR deck, and "puts" it into the VCR to record the TV program (above-right).

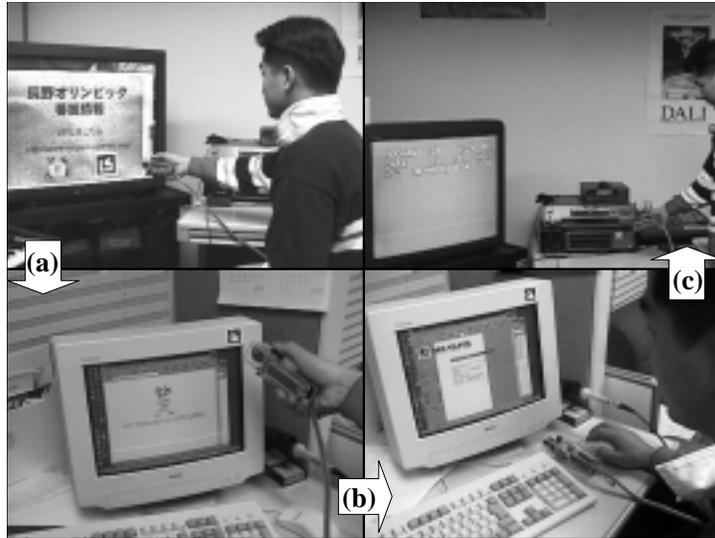


Fig. 8. (a) Picking up a URL from a TV screen and dropping it into a computer, (b) browsing TV program information on the computer, and (c) operating a VCR by picking up the TV program information from the computer and dropping it in the VCR deck

4 Implementation Details

4.1 Hardware Architecture

A prototypical InfoStick is depicted in Figure 9. We made it as compact as possible to allow the user to operate it with only one hand. The InfoStick consists of the following items: a SONY CCD-MC1 as a CCD camera to sense video images from a physical environment, three input buttons to “get”/“put”/“select”, and a SUNLIKE 16×2 LCD as a display to show the particular information that the InfoStick can “get”/“put”/“select”. A PARALLAX BASIC STAMP II is included in the InfoStick as a small computer for controlling the I/O signal and LCD and for exchanging data between the InfoStick and a Mitsubishi AMiTY-CN with RS-232 serial communication protocols. Parallax BASIC Stamps are small computers that run Parallax BASIC (PBASIC) programs. They have programmable I/O pins that can be used to directly interface to TTL-level devices, such as buttons, LCDs, speakers, and shift registers. And with extra components, these I/O pins can be connected to non-TTL devices, such as RS-232 networks. The Mitsubishi AMiTY-CN is located on the user’s waist. It is a mobile computer for recognizing the 2D matrix code from the CCD camera’s video image, by using an IBM Smart Capture Card II, and for communicating with other computers on the network. For electric appliances that are not on the network, the nearest IBM PC communicates with other PCs and controls the electric appliances

through them. SONY VboxII-CI1100s connected to the IBM PCs control electric appliances.

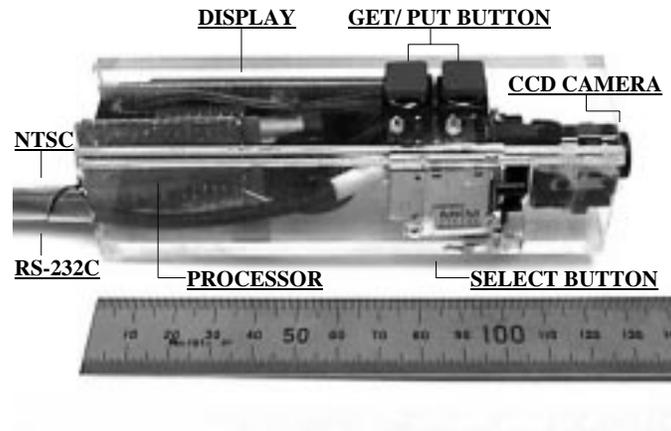


Fig. 9. Configuration of the InfoStick

4.2 Software Architecture

Figure 10 illustrates the structure of the software in the InfoStick system. A video image from the CCD camera is first sent to the Marker Reader. The Marker Reader then extracts the black-and-white pattern of the image that the InfoStick is pointing at and analyzes it. If a 2D Matrix Code is found from the image, it is converted to an ID number that will be sent to the InfoStick Controller. Then the controller receives the ID number, it searches for the target object's name and the corresponding available information that the InfoStick can "get"/"put" from the InfoStick DataBase. Marker Reader and InfoStick Controller are implemented in the InfoStick, and the other software modules are in computers connected to the network. The InfoStick DataBase contains the information received by the InfoStick and the ID Table corresponding to the ID number given by the Information Manager on the network. If the user wants to operate one of objects, he can "select" and "get"/"put" the necessary information with the input buttons. In the case of "get", the target information is added into the InfoStick DataBase. On the other hand, in the case of "put", the target physical object is operated by the Machine Controller with the necessary information obtained from the Information Manager.

All code without PBASIC is written in Java and executed on an IBM PC running Windows95. The video capturing class is of special note as it uses

JDK1.1 Java-native-interface (JNI). The Information Manager and ID table are also Java applications and communicate with other experimental applications through TCP/IP connections. All physical objects used in our applications are directly connected to the Ethernet. The InfoStick, however, uses a wireless local area network. A user therefore can take the InfoStick anywhere within wireless coverage.

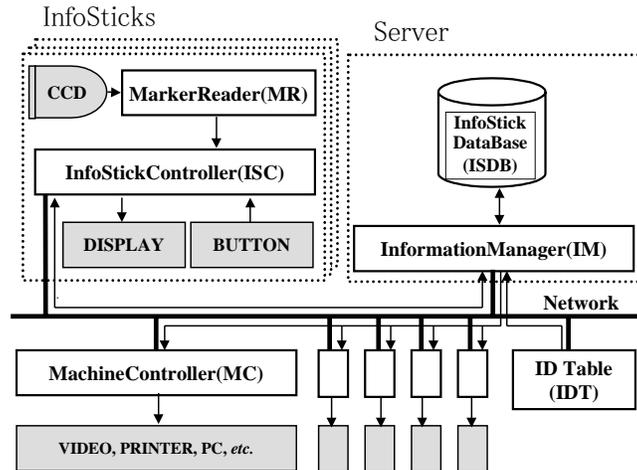


Fig. 10. Software architecture of the InfoStick system

5 Related Work

There are several hand-held devices that are designed for operating other digital and physical objects. Most of them, however, aim to operate a single target (e.g., a remote controller for a TV set), rather than to operate several target objects.

The PDA-ITV [4] uses a PDA as a commander for interactive TV. Although it uses two different displays for one task, the roles of PDA and TV are static; PDA acts as a commander only for the TV. Neither seamless manipulation is possible, nor exchanging information between the PDA and ITV is interactive. For example, it is not possible to “pick up” information from the TV screen, and then “drop” it into the PDA. The PaperLink [1] is a computer augmented pen with a video camera that is capable of recognizing text on a printed document. Although PaperLink can pick up information from paper and put it on other paper, it does not support inter object operations. For example, the user can not operate a computer object and paper information with the same PaperLink pen.

MediaBlocks [5] is a small tag used as a physical container of digital information. The user can virtually attach and carry digital data by using this tag. This system assumes every information appliance has a tag reader/writer, making it difficult to scale this environment. A user is unable to see the carried data until the tag is actually inserted into a tag reader/writer. Finally, the Pick-and-Drop system [2] is a direct manipulation technique that can be used to transfer data between different computers as well as on the same computer. Pick-and-Drop allows the user “pick up” an object from a display and drop it onto another display as if he were operating a physical object. Our InfoStick system is an extension of this system. Although our system also uses the Pick-and-Drop metaphor for “get” and “put” operations, the purpose of the InfoStick is to operate not only computers but all physical objects.

6 Conclusion and Future Work

We have developed the InfoStick, a new hand-held interaction device that aims to provide a uniform way of operating everyday digital/physical objects. Currently, the InfoStick recognizes the target object by using a combination of attached visual markers and a video camera. When it identifies the ID of the object, the InfoStick searches for the object’s name and available information related to that name from the InfoStick DataBase on the network. The idea of deploying visual markers for object-level identification in the environment is aging. However, the use of these markers has advantages and disadvantages. This technique enables to identify an object even if it is not connected to the network, such as a printed material like a poster, a book, or a newspaper, and that visual marker can operate it with easy technology at low cost. On the other hand, a user must always know where the visual marker is. And the camera in the InfoStick needs to be pointed at it.

There are other ways of recognizing objects. They include wireless tags and infrared (IR) beacons. In the case of using a wireless tag, when a receiver approaches within about one meter of the tag, the receiver can recognize its ID number. On the other hand, the IR beacon transmits the ID number to the environment periodically. This beacon covers room-size area and is relatively robust regarding orientation of the sensors. For each method for recognizing objects, there is a different advantage and by using it appropriately, the InfoStick will become more widely used.

In addition, we are planning to provide a “docking station” of the InfoStick for easy information exchange with PCs. When a user “docks” the InfoStick, information stored in the InfoStick is transferred to the PC and the user can exchange them on the PC window with the mouse. The mouse’s advantages include being able to “grab”, “drag” and “drop” one object from many objects on the same window. On the contrary, when the InfoStick receives a lot of information, it is difficult to select items on the LCD display. If the InfoStick icon pops up on a window when it is put into a docking station connected to a computer, we can release, delete and select objects on the window by using the mouse. (Figure 11).

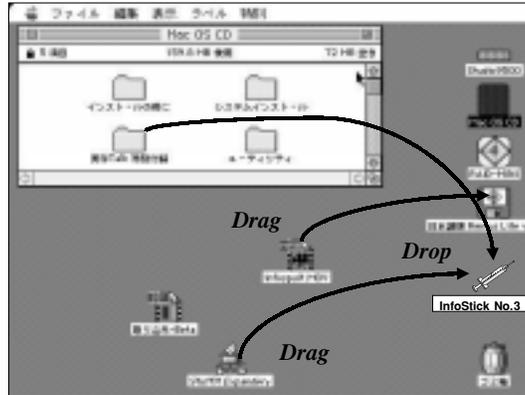


Fig. 11. Drag-and-drop to the InfoStick icon on the computer window

Acknowledgements

We wish to thank Jun Yamamoto, Nobuyuki Matsushita and Masanori Saitoh from Keio University for their contributions to this project. Several helpful discussions with members of the Sony CSL real-world UI group were also very helpful. We would also like to express our appreciation to Mario Tokoro and Toshi Doi for supporting this research.

References

1. Toshifumi Arai, Dietmar Aust, and Scott Hudson. Paperlink: A technique for hyperlink from real paper to electronic content. In *Proceedings of CHI'97*, pp.327–333, 1997.
2. Jun Rekimoto. Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments. In *Proceedings of UIST'97*, pp. 31–39, 1997.
3. Jun Rekimoto, and Masanori Saitoh. Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments. In *Proceedings of CHI'99*, pp. 378–385, 1999.
4. Stott Robertson, Cathleen Wharton, Cathsrine Achworth, and Marita Franzke. Dual device user interface design: PDAs and interactive television. In *Proceedings of CHI'96*, pp. 79–86, 1996.
5. Brygg Ullmer, Hiroshi Ishii, and Dylan Glas. mediaBlocks: Physical Containers, Transports, and Control for Online Media, In *Proceedings of SIGGRAPH'98*, pp. 379–386, 1998.
6. Mark Weiser. The computer for the twenty-first-century. *Scientific American*, pp.94–104, 1991.

Tiles: A Mixed Reality Authoring Interface

Ivan Poupyrev^{1,i}, Desney Tan^{2,i}, Mark Billinghurst³, Hirokazu Kato^{4,6},
Holger Regenbrecht⁵ & Nobuji Tetsutani⁶

¹Interaction Lab, Sony CSL
3-14-13 Higashi-Gotanda,
Tokyo 141-0022
Japan

²School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue,
Pittsburgh, PA 15213, USA

³University of Washington
⁴Hiroshima City University
⁵DaimlerChrysler AG
⁶ATR MIC Labs

poup@csl.sony.co.jp, desney@cs.cmu.edu, grof@hitl.washington.edu,
kato@sys.im.hiroshima-cu.ac.jp, holger.regenbrecht@daimlerchrysler.com,
tetsutani@mic.atr.co.jp

Abstract: Mixed Reality (MR) aims to create user interfaces in which interactive virtual objects are overlaid on the physical environment, naturally blending with it in real time. In this paper we present *Tiles*, a MR authoring interface for easy and effective spatial composition, layout and arrangement of digital objects in MR environments. Based on a tangible MR interface approach, *Tiles* is a transparent user interface that allows users to seamlessly interact with both virtual and physical objects. It also introduces a consistent MR interface model, providing a set of tools that allows users to dynamically add, remove, copy, duplicate and annotate virtual objects anywhere in the 3D physical workspace. Although our interaction techniques are broadly applicable, we ground them in an application for rapid prototyping and evaluation of aircraft instrument panels. We also present informal user observations and a preliminary framework for further work.

Keywords: Augmented and mixed reality, 3D interfaces, tangible and physical interfaces, authoring tools

1 Introduction

Virtual objects are pervading our living and working environments, augmenting and even replacing physical objects. Electronic billboards are starting to replace familiar paper billboards in public spaces; and signs providing directions are often projected, rather than made out of the physical plastic or paper.

Mixed Reality research takes this integration between physical and virtual worlds even further. MR systems create advanced user interfaces and environments where interactive virtual objects are overlaid on the 3D physical environment, naturally blending with it in real time (Azuma, 1997; Milgram, Takemura, Utsumi, *et al.*, 1994). There are many potential uses for such interfaces, ranging from industrial, to medical and entertainment applications (e.g. Bajura, Fuchs *et al.* 1992; Poupyrev, Berry *et al.* 2000, see also Azuma, 1997 for survey).

In our work, we are interested in applying MR techniques to the task of collaborative design (Fjeld, Voorhorst, Bichsel, *et al.*, 1999; Kato, Billinghurst, Poupyrev, *et al.*, 2000). In one scenario, several ar-

chitects and city planners gather around a conventional physical model of the city to evaluate how proposed buildings would alter the city appearance. Instead of using physical models of new buildings, the participants manipulate virtual 3D graphics models that are correctly registered and superimposed on the physical city model. The new buildings are virtual, so they can be quickly altered on the fly, allowing designers to evaluate the alternatives and possible solutions. Dynamic simulations, such as traffic flow and pollution can be simulated and superimposed right on the physical city model.

Unlike virtual reality (VR) interfaces, MR do not remove users from their physical environment. Users still have access to conventional tools and information, maps, and design schemes. Users can also continue to see each other and use gestures or facial expressions to facilitate their communication and enhance the decision process. Furthermore, as they proceed with their discussion they are implicitly documenting the design process by marking and annotating both virtual and physical objects.

This scenario remains mostly hypothetical. Most current MR interfaces work as information browsers allowing users to see virtual information embedded

¹ This work was conducted while the author was working at the ATR MIC Labs, Japan

into the physical world. However, few provide tools that let the user interact, request or modify this information effectively and in real time (Rekimoto, et al. 1998). Even the basic interaction tasks and techniques, such as manipulation, coping, annotating, dynamically adding and deleting virtual objects to the MR environment have been poorly addressed.

The current paper presents *Tiles*, a MR authoring interface that investigates interaction techniques for easy and effective spatial composition, layout and arrangement of digital objects in mixed reality environments. Several features distinguish *Tiles* from previous work. First, *Tiles* is a transparent interface that allows seamless two-handed 3D interaction with both virtual and physical objects. *Tiles* does not require participants to use or wear any special purpose input devices, e.g. magnetic 3D trackers, to interact with virtual objects. Instead users can manipulate virtual objects using the same input devices they use in physical world – their own hands. Second, unlike popular table-top based AR interfaces, where the virtual objects are projected on and limited by the 2D surface of a table (e.g. Rekimoto and Saitoh, 1999), *Tiles* allows full 3D spatial interaction with virtual objects anywhere in their physical workspace. The user can pick up and manipulate virtual data just as real objects, as well as arrange them on any working surface, such as a table or whiteboard. Third, *Tiles* allows the user to use both digital and physical annotations of virtual objects, using conventional tools such as PostIt™ notes. Finally, in *Tiles* we attempt to design a simple yet effective interface for authoring MR environments, based on a consistent interface model, providing a set of tools that allows users to add, remove, copy, duplicate and annotate virtual objects in MR environments. Although 2D and 3D authoring environments have been one of the most intensively explored topics in desktop and VR interfaces (e.g. Butterworth, Davidson, Hench, et al., 1992; Mapes and Moshell, 1995) there are far fewer attempts to develop authoring interfaces for mixed reality. We discuss some of them in the next section.

2 Related work

We spend a significant part of our everyday life arranging and assembling physical objects in our workspace: books, papers, notes and tools. In recent years there has been a trend towards developing computer interfaces that also use physical, tangible objects for input devices. For example, in the Digital Desk project (Wellner, 1993), the position of paper documents and the user's hands on an augmented table were tracked using computer vision techniques. In this system, the user could seamlessly arrange and annotate both real paper and virtual documents using the same physical tool – a conventional pen. This approach was extended with graspable and tangible

interfaces, which have been proposed as a possible interface model for such environments. This idea suggests using simple physical objects tracked on the surface of a table as either physical handles allowing to select, translate and rotate electronic objects or as data transport devices (Fitzmaurice, Ishii and Buxton, 1995; Fjeld, et al., 1999; Ishii and Ullmer, 1997; Ullmer and Ishii, 1997; Ullmer, Ishii and Glas, 1998). Alternatively, Rekimoto, et al. (1999) used a special purpose laser pointer device and Hyperdragging interaction technique to move electronic documents between the computer and a shared workspace.

The main advantage of this approach is that the user does not have to wear any special-purpose display devices, such as a head-mounted display (HMD). Furthermore, physical, tangible interfaces allow the user to seamlessly interact with both electronic and physical objects simply with hands and physical tools, e.g. pen and wood blocks. However, because the output is limited to the 2D surface of the table, the user is not able pick up virtual documents and manipulate them freely in space as can be done with real paper documents. This interaction is also limited to flat paper-like objects. Presentation and manipulation of 3D virtual objects in such environments, though possible, is difficult and inefficient (Fjeld, et al., 1999). Hence, these interfaces introduce *spatial seams*¹ in mixed reality environments – the interfaces are localized on an augmented surface and cannot extend beyond it.

Another fundamental alternative approach to building mixed reality workplaces is three-dimensional Augmented Reality (AR) (Azuma, 1997). In this approach, virtual objects are registered in 3D physical environments using magnetic or computer vision tracking techniques and then presented to the user looking through a HMD (e.g. Bajura, et al., 1992; Feiner, MacIntyre and Seligmann, 1993) or a handheld display device (e.g. Fitzmaurice, 1993; Rekimoto and Nagao, 1995). Unlike tabletop-based MR, this approach allows the system to render 3D virtual objects anywhere in the physical environment to provide spatially seamless MR workspaces.

However, as Ishii points out, most AR researchers are primarily concerned with “considering purely visual augmentations” rather than the interaction and physical context of AR environments (Ishii and Ullmer, 1997). This has led to difficulty with designing interaction techniques that would let the user effectively manipulate 3D virtual objects distributed freely in a 3D workspace. Previous approaches to solve this problem include using a special purpose 3D input device to select and manipulate virtual ob-

¹ Ishii defines a seam as a discontinuity or constraint in interaction that forces the user to shift among a variety of spaces or modes of operation (Ishii, Kobayashi and Arita, 1994).

jects, such as magnetic trackers used in Studierstube (Schmalsteig, Fuhrmann, Szalavari, *et al.*, 1996) and MARS systems (Hollerer *et al.* 1999). Traditional input devices, such as a hand-held mouse or tablet (Hollerer, *et al.*, 1999; Rekimoto, *et al.*, 1998), as well as speech input and intelligent agents (Anabuki, Kakuta, Yamamoto, *et al.*, 2000) have also been investigated. The major disadvantage with these approaches is that the user is forced to use two different interfaces – one for the physical and one for the virtual objects. Thus, the natural workflow is broken with *interaction seams* – every time the user needs to manipulate virtual objects, he or she needs to use a special purpose input device that would not be normally used in real world interaction.

Thus the current design of mixed reality interfaces, falls into two orthogonal approaches: tangible interfaces and tabletop MR offer seamless interaction but results in spatial discontinuities, while 3D AR provides spatially seamless mixed reality workspaces but introduces discontinuities in interaction. This paper presents an approach that merges the best qualities of both interaction styles. The *Tiles* system was developed to provide true spatial registration and presentation of 3D virtual objects anywhere in the physical environment. At the same time we implement a tangible interface that allows users to interact with 3D virtual objects without using any special purpose input devices. Since this approach combines tangible interaction with AR display we refer to it as *Tangible Augmented Reality*. In the next section we show how the *Tangible AR* can be used to build a simple yet effective MR authoring interface.

3 Tiles Interface

Tiles is a collaborative *Tangible AR* interface that allows several participants to dynamically layout and arrange virtual objects in a mixed reality workspace. In this system, the user wears a light-weight head-mounted display (HMD) with a small camera attached, both of which are connected to a computer. Output from the camera is captured by the computer which then overlays virtual images onto the video in real time. The resulting augmented view of the real world is then presented back to the user on his or her HMD so the user sees virtual objects embedded in the physical workspace (Figure 1 and Figure 2). The 3D position and orientation of virtual objects is determined using computer vision tracking techniques, tracking 3D position and orientation of square fiducial markers that can be attached to any physical object. The tracking techniques have been inspired by Rekimoto (1988) and are more completely described in (Kato and Billinghurst, 1999) The virtual objects are rendered relative to these markers, and by manipulating marked physical objects, the user can

manipulate virtual objects without need to use any additional input devices.

The rest of this section presents the *Tiles* interface and interaction techniques. Although our interface techniques are broadly applicable, the *Tiles* system has been developed for rapid prototyping and evaluation of aircraft instrument panels, a joint research initiative carried out with support from DASA/EADS Airbus and DaimlerChrysler AG. To ground further discussion and illustrate the rationale for our design decisions, we present a brief overview of the application design requirements.

3.1 Design Requirements

The design of aircraft instrument panels is an important procedure that requires the collaborative efforts of engineers, human factor specialists, electronics designers, airplane pilots and many others. Because mistakes are normally detrimental to aircraft safety, designers and engineers are always looking for new technologies that can reduce the cost of designing, prototyping, and evaluating the instrumental panels without compromising design quality. Since they are often building upon existing functional instruments, designers have taken a special interest in MR interfaces. This is because they often need to evaluate prototypes of instruments relative to existing instrumental panels, without having to physically build them. This design activity is inherently collaborative and involves team-based problem solving, discussions and joint evaluation. It also involves heavy use of existing physical plans, documents and tools.

Using observations of how instrument panels are currently designed, DASA/EADS Airbus and DaimlerChrysler engineers produced a set of requirements for MR interfaces to support this task. They envisioned MR interfaces allowing groups of designers, engineers, human factors specialists, and aircraft pilots to collaboratively outline and layout a set of virtual aircraft instruments on a board simulating an airplane cockpit. Designers would need to be able to easily add and remove virtual instruments from the board using a catalog of the virtual instruments. After the instruments are placed on the board, they would like to evaluate and rearrange the position of the instruments as necessary. The interface should also allow the use of existing physical schemes and documents with conventional tools, e.g. whiteboard markers, to let participants document solutions and problems, as well as add physical annotations to virtual instruments. A further requirement was that the resulting interface be intuitive, easy to learn and use.

3.2 Interface

3.2.1 Basics: Tiles interface components

The *Tiles* workspace and interface consist of: 1) a metal whiteboard in front of the user; 2) a set of

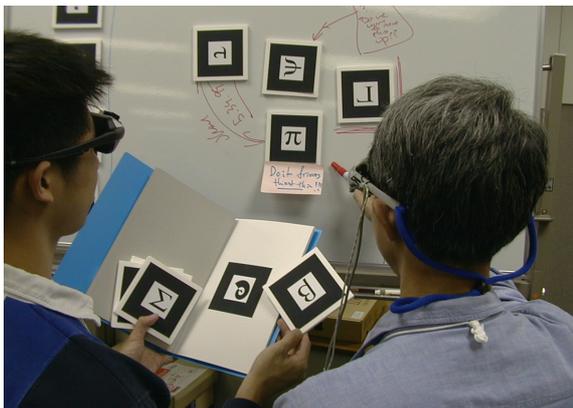


Figure 1: *Tiles* environment: users collaboratively arrange data on the whiteboard, using tangible data containers, *data tiles*, as well as adding notes and annotations using traditional tools: whiteboard pen and notes.

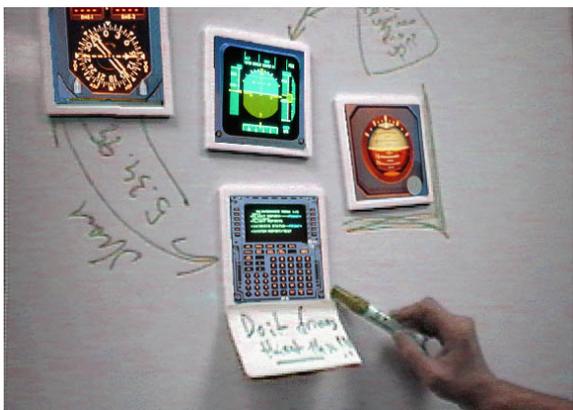


Figure 2: The user, wearing lightweight head-mounted display with mounted camera, can see both virtual images registered on tiles and real objects.

cardboard cards (15 by 15 centimetres each) with tracking patterns attached to them, which we call *tiles*. Each of these cards has a magnet on the back so it can be placed on and removed from the whiteboard; 3) a book, with marked pages, which we call *book tiles*, and 4) conventional tools used in discussion and collaboration, such as whiteboard pens and PostIt™ notes (Figure 1 and Figure 2).

The whiteboard acts as a shared collaborative workspace, where users can rapidly draw rough layout of virtual instruments using whiteboard markers, and then visualize this layout by placing and arranging tiles with virtual instruments on the board.

The tiles act as generic tangible interface controls, similar to icons in a GUI interface. So instead of interacting with digital data by manipulating icons with a mouse, the user interacts with digital data by physically manipulating the corresponding tiles. Although the tiles are similar to physical icons (phicons), introduced in metaDesk system (Ullmer and Ishii, 1997), there are important differences. In metaDesk, the authors proposed a close coupling

between physical properties of phicons, i.e. their shape and appearance, to virtual object that phicons represent. For example, the shape of phicons representing a certain building had an exact shape of that particular building. In designing the *Tiles* interface we attempted to decouple physical properties of tiles from the virtual data as much as possible – the goal was to design universal data containers that can hold *any* digital data or no data at all. Interaction techniques for performing basic operations such as putting data on tiles and removing data from tiles are the same for all tiles, resulting in a consistent and streamlined user interface. This is not unlike GUI interfaces, where all basic operations on icons are the same irrespective of whether they represent a document or a game program – i.e. the user can move, open, resize and delete icons. Furthermore, because the user can dynamically put any digital data on the tile, our system does not require an excessive number of tiles, since they can be “recycled”.

3.2.2 *Classes of tiles: data, operators and menu*

Not all tiles are the same – we use three classes of tiles: *data tiles*, *operator tiles* and *menu tiles*. All tiles share similar physical appearances and common operation. The only difference in their physical appearance is the icons identifying tile types. This allows users who are not wearing a HMD to identify the tiles purpose. Below we briefly summarize the basic properties of each of the classes:

- *Data tiles* are generic data containers. The user can put and remove virtual objects from the data tiles; if a data tile is empty, nothing is rendered on it. We use Greek symbols as tracking patterns to identify the data tiles.
- *Operator tiles* are used to perform basic operations on data tiles. Currently implemented operations include *deleting* a virtual object from a data tile, *copying* a virtual object to the clipboard or from clipboard to the data tile, and requesting *help* or annotations associated with a virtual object on the data tile. Iconic patterns are used to identify each operator tile, for example the tile that deletes a virtual object from data tiles is identified with a trashcan icon. In MR the operator tiles are also identified by virtual 3D widgets attached to them.
- *Menu tiles* make up a book with tiles attached to each page (Figure 1). This book works like a catalogue or a menu: as the user flips through the pages, he can see virtual objects attached to each page, choose the required instrument and then copy it from the book to any empty data tile.

3.2.3 *Operations on tiles*

All tiles can be manipulated in space and arranged on the whiteboard: the user simply picks up any of

the tiles, examines its contents and places it on the whiteboard. Operations *between tiles* are invoked by bringing two tiles next to each other (within a distance less than 15% of the tile size).

For example, to copy an instrument to the data tile, the user first finds the desired virtual instrument in the menu book and then places any empty data tile next to the instrument (Figure 7). After a one second delay to prevent an accidental copying, a copy of the instrument smoothly slides from the menu page to the tile and is ready to be arranged on the whiteboard. Similarly, if the user wants to “clean” data from tile, the user brings the trashcan tile close to the data tiles, removing the instrument from it (Figure 3).

Using the same technique we can implement copy and paste operations using the *clipboard operator*: the user can copy an instrument from any of the data tiles to the clipboard and then from clipboard to an empty data tile (Figure 4). The current content of the clipboard is always visible on the virtual clipboard icon. There can be as many clipboards as needed – in the current implementation we have two independent clipboards.



Figure 3: The user cleans data tiles using trash can operator tile. The removed virtual instrument is animated to provide the user with smooth feedback.



Figure 4: Copying data from clipboard to an empty data tile.

Table 1 summarises the allowed operations between tiles. Note that we have not defined any operations between data tiles because this would cause interaction between data tiles and not allow the user to lay them next to each other on the whiteboard.

3.2.4 Getting help in Tiles

Help systems have been one of the corner stones in providing guidance to users in a GUI, and effective MR interfaces will also require effective on-line help facilities. Therefore, we implemented a help tile: to receive help on any virtual object, the user simply places the help tile next to the data tile on which they require help. In the simplest case, this triggers explanatory text that appears within a bubble next to the help icon (Figure 5). Currently, this function is used by the designer to leave short digital annotations on the virtual instruments and to provide help for users while they manipulate the operator tiles.

3.2.5 Mixing physical and virtual tools in Tiles

The *Tiles* interface allows the users to seamlessly combine use of conventional physical tools, such as whiteboard pens, together with the virtual tools that we introduced in the previous sections. For example, the user can physically annotate a virtual aircraft instruments using a standard whiteboard pen or sticky note (see Figure 1, 2 and 6).

3.2.6 Collaboration

Tiles has been designed with collaboration in mind and allows several users interact in a same augmented workspace. We have been evaluating two possible scenarios: 1) All users are equipped with HMDs and can directly interact with virtual objects (Figure 1) and 2) Non-immersed users, i.e. users that do not wear HMDs collaborate with immersed users using an additional monitor presenting the view of immersed collaborator (Figure 7).

2.1 Initial User Feedback

Although the *Tiles* system has not yet been evaluated in rigorous user studies we have presented the interface in several public settings and received informal feedback from typical users. The *Tiles* system was first demonstrated at the IEEE/ACM International Symposium for Augmented Reality (ISAR) 2000 in Munich, Germany. About seventy users tested the system. We observed that with simple instructions, most of these users were able to quite effectively simulate the design process, laying out and rearranging the instruments on the board. They found the system easy to use, intuitive and quite enjoyable. DaimlerChrysler design engineers found that the concept meets the basic requirement for the authoring of MR environments and thought it promising enough to start evaluating its feasibility in real industrial applications.

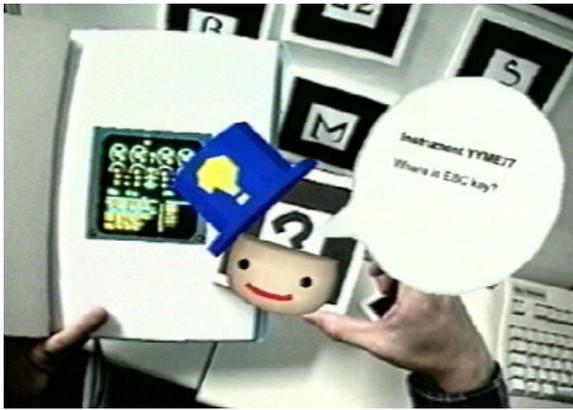


Figure 5: The user invokes an electronic annotation attached to the virtual objects using the help tile



Figure 6: Physically annotating virtual objects in *Tiles*



Figure 7: Collaboration between immersed and non-immersed users in *Tiles* environment

The most prevalent complaint was the physical design of the tiles. In designing the system, we wanted to keep the physical tiles as small as possible so as to match the size of the actual instruments. However, we tried to make the markers large enough for reliable tracking. As a result, the border around the tracked area, on which the user could place their fingers when holding the card, was uncomfortably small. Furthermore, the users tended to occlude the

tracking border, which resulted in tracking failure. We are currently exploring different physical designs for the tiles in the next version of the system.

Our initial experiments with the non-immersed collaboration mode was encouraging in that the users were able to collaborate rather effectively. All interface components are simple physical objects identified with graphical icons, so the non-immersed user was able to perform the same authoring tasks as immersed user, i.e. laying out the tiles on the whiteboard, evaluating it, copying the virtual instruments on the data tiles and etc. We are planning to perform more extensive studies of this collaboration mode.

2.2 Implementation

The fundamental elements of any MR systems are techniques for tracking user position and/or view-point direction, registering virtual objects relative to the physical environment, rendering, and presenting them to the user.

The *Tiles* system is implemented using ARTool-Kit, a custom video see-through tracking and registering library (Kato and Billinghurst, 1999). We mark 15x15 cm paper cards with simple square fiducial patterns consisting of thick black border and unique symbols in the middle identifying the pattern. The system does not have restrictions on symbols used for identification as long as it is asymmetrical to distinguish between the 4 possible orientations of the square border. The user wears a Sony Glasstron PLMS700 headset, which is lightweight and comfortable and provides VGA 800 by 600 pixel resolution. This was sufficient for reading text images rendered in our MR environment. A miniature NTSC Toshiba camera with a wide-angle lens (2.2 mm) is attached to the headset. The video stream from the camera is captured at 640x240 resolution to avoid interlacing problems and scaled back to 640x480 by using a line doubling technique.

After the computer vision pattern tracking identifies localization marks in the video stream, the relative position and orientation of the marks relative to the head-mounted camera can be determined and virtual objects can then be correctly rendered on top of the physical cards. Although the wide-angle lens distorts the video image, our tracking techniques are robust against these distortions and able to correctly track patterns without losing performance.

All virtual objects are represented as VRML97 models and a custom VRML browser has been built to manipulate and render 3D objects into the video stream. In the current *Tiles* application the system tracks and recognize 21 cards in total. The software is running on an 800Mhz Pentium III PC with 256Mb RAM and the Linux OS. This produces tracking and display rate of between 25 and 30 frames per second.

Operation	Result
Menu operations	
 +  = 	
Clipboard operations	
 +  = 	
 +  = 	
 +  = 	
Trashcan operations	
 +  = 	
 +  = Not defined	
 +  = 	
Help operations	
 +  = 	
 +  = 	
 +  = Not defined	

Table 1: Operations defined for different tiles types: e.g. bringing together menu tile and empty data tile will move instrument on the tile (first row in the table).

4 Discussion and Future Work

The *Tiles* system is a prototype tangible augmented reality authoring interface that allows a user to quickly layout virtual objects in a shared workspace and easily manipulate them without need of special purpose input devices. We are not aware of any previous interfaces that share these properties. In this section we discuss some of the *Tiles* design issues and future research directions.

Generality of Tiles, other applications. The interface model and interaction techniques introduced in *Tiles* can be easily extended to other applications that require mixed reality interfaces. Object modification techniques, for example, can be quite easily introduced into *Tiles* by developing additional operator cards that would let the user dynamically modify

objects, e.g. scale them, change their colour and so on. We are also currently exploring more direct techniques that would track users' hands and allow the user to touch and scale virtual objects directly with gestures.

Although developing additional interaction techniques would allow *Tiles* to be used in many different application scenarios, we should note that in MR environments the user can easily transfer between the MR workspace and a traditional environments such as a desktop computer. Therefore, we believe that the goal of developing MR interfaces is not to bring every possible interaction tool and technique into the MR workspace, but to balance and distribute the features between the MR interface and other media: some tools and techniques are better for MR, some are better to be left for traditional tools. Hybrid mixed reality interfaces have been suggested by a number of researchers and are an interesting and important research direction (Schmalstieg, Fuhrmann and Hesina, 2000)

Ad-hoc, re-configurable interfaces. An interesting property of mixed reality interfaces is their ad-hoc, highly re-configurable nature. Unlike the traditional GUI and 3D VR interfaces, where the interface layout is mostly determined by an interface designer in advance, the MR interfaces are in some sense designed by user as they are carrying on with their work. Indeed, in *Tiles* the users are free to put interface elements anywhere they want: tables, whiteboards, in boxes and folders, arrange them in stacks or group them together. How the interface components should be designed for such environments, if they should be aware of the dynamic changes in their configuration, and how this can be achieved are interesting research directions.

Physical form-factor. Our initial user observations showed that in designing tangible MR interfaces, the form factor becomes an important design issue. Indeed, the main problem reported with *Tiles* was that the cards were too small, so people tended to occlude the tracking markers. In MR interfaces both the physical design of the interfaces and the computer graphics design of virtual icons attached to the interfaces is important. The design of physical components can convey additional semantics of the interface, for example the shape of the physical cards can be designed so that they can snap into each other as pieces in a jigsaw puzzle, and depending on their physical configuration resulting functionality of the interface could be different. Expressing different interface semantics by explicitly using the shape of the interface components can also be explored further in *Tiles* environment.

Remote and face-to-face collaboration. The current *Tiles* interface provides only very basic collaborative capabilities for co-located users. We are plan-

ning to explore remote collaboration techniques in the *Tiles* interface by using a digital whiteboard and global static camera to capture the writings on the whiteboard and location of tiles, and then distribute this to remote participants.

5 Conclusions

In this paper we presented *Tiles*, a MR authoring interface for easy and effective spatial composition, layout and arrangement of digital objects in MR environments. Based on a tangible MR interface approach, *Tiles* is a transparent user interface that allows users to seamlessly interact with both virtual and physical objects and introduces a consistent MR interface model, providing users a set of tools that allow dynamically to add, remove, copy, duplicate and annotate virtual objects anywhere in the 3D physical workspace. Although our interaction techniques are broadly applicable, we grounded them in an application for rapid prototyping and evaluation of aircraft instrument panels, a joint research initiative carried out with support from DASA/EADS Airbus. Informal user observations were encouraging and a framework for further work has been outlined.

References

- Anabuki, M., Kakuta, H., Yamamoto, H., Tamura, H. (2000). Welbo: An Embodied Conversational Agent Living in Mixed Reality Spaces. In *Proceedings of the CHI'2000, Extended Abstracts*, ACM, pp. 10-11.
- Azuma, R. (1997). A Survey of Augmented Reality. *Presence*, 6(4), pp. 355-385.
- Bajura, M., Fuchs, H., Ohbuchi, R. (1992). Merging Virtual Objects with the Real World: Seeing Ultrasound Imagery Within the Patient. In *Proceedings of the SIGGRAPH'92*, ACM, pp. 203-210.
- Butterworth, J., Davidson, A., Hench, S., Olano, T. (1992). 3DM: a three dimensional modeler using a head-mounted display. In *Proceedings of the Symposium on Interactive 3D graphics*, ACM, pp. 135-138.
- Feiner, S., MacIntyre, B., Seligmann, D. (1993). Knowledge-Based Augmented Reality. *Communications of the ACM*, 36(7), 53-62.
- Fitzmaurice, G., Ishii, H., Buxton, W. (1995). Bricks: Laying the foundations for graspable user interfaces. In *Proceedings of the CHI'95*, ACM, pp. 442-449.
- Fitzmaurice, G. W. (1993). Situated information spaces and spatially aware palmtop computers. *Communication of the ACM*, 36(7), 38-49.
- Fjeld, M., Voorhorst, F., Bichsel, M., Lauche, K., Rauterberg, M. et al. (1999). Exploring Brick-Based Navigation and Composition in an Augmented Reality. *Proceedings of the HUC99*, pp. 102-116.
- Hollerer, T., Feiner, S., Terauchi, T., Rashid, G., et al. (1999). Exploring MARS: developing indoor and outdoor user interfaces to a mobile augmented reality system. *Computers & Graphics*, 23, pp. 779-785.
- Ishii, H., Kobayashi, M., Arita, K. (1994). Iterative design of seamless collaborative media. *CACM* 37(8), 83-97.
- Ishii, H., Ullmer, B. (1997). Tangible bits towards seamless interfaces between people, bits and atoms. In *Proceedings of the CHI'97*, ACM pp. 234-241
- Kato, H., Billinghurst, M. (1999). Marker Tracking and HMD Calibration for a Video-based AR Conferencing System, *2nd Int. Workshop on AR*, pp.85-94.
- Kato, H., Billinghurst, M., Poupyrev, I., Imamoto, K., Tachibana, K. (2000). Virtual Object Manipulation on a Table-Top AR Environment, *ISAR*, pp. 111-119
- Mapes, D., Moshell, J. (1995). A Two-Handed Interface for Object Manipulation in Virtual Environments. *Presence*, 4(4), pp. 403-416.
- Milgram, P., Takemura, H., Utsumi, A., Kishino, F. (1994). Augmented Reality: A Class of Displays on the Reality-Virtuality Continuum, *SPIE*, 2351
- Poupyrev, I., Berry, R., Kurumisawa, J., Nakao, K., Billinghurst, M., Airola, C., Kato, H., et al. (2000). Augmented Groove: Collaborative Jamming in Augmented Reality. *SIGGRAPH'2000 CA&A*, pp. 77.
- Rekimoto, J. (1988). Matrix: A Realtime Object Identification and Registration Method for Augmented Reality. In *Proceedings of APCHI'98*. ACM
- Rekimoto, J., Ayatsuka, Y., Hayashi, K. (1998). Augment-able reality: Situated communication through physical and digital spaces. In *Proc. ISWC'98*. IEEE.
- Rekimoto, J., Nagao, K. (1995). The World through the Computer: Computer Augmented Interaction with Real World Environments. In *Proceedings of the UIST'95*, ACM, pp. 29-36
- Rekimoto, J., Saitoh, M. (1999). Augmented surfaces: A spatially continuous work space for hybrid computing environments. In *Proc. CHI'99*, ACM, pp. 378-385.
- Schmalstieg, D., Fuhrmann, A., Szalavari, Z., Gervautz, M. (1996). Studierstube - An Environment for Collaboration in Augmented Reality. *CVE '96 Workshop*
- Schmalstieg, D., Fuhrmann, A., Hesina, G. (2000). Bridging multiple user interface dimensions with augmented reality systems. *ISAR'2000*, IEEE, pp. 20-29.
- Ullmer, B., Ishii, H. (1997). The metaDesk: Models and Prototypes for Tangible User Interfaces. In *Proceedings of the UIST'97*, ACM, pp. 223-232.
- Ullmer, B., Ishii, H., Glas, D. (1998). mediaBlocks: Physical containers, transports and controls for online media. *SIGGRAPH'98*, ACM, pp. 379-386.
- Wellner, P. (1993). Interaction with paper on the digital desk. *Communications of the ACM*, 36(7), pp. 87-96.

Virtual Object Manipulation on a Table-Top AR Environment

H. Kato¹, M. Billinghurst², I. Poupyrev³, K. Imamoto¹, K. Tachibana¹

¹*Faculty of Information Sciences,
Hiroshima City University
3-4-1, Ozuka-higashi, Asaminami-ku,
Hiroshima, 731-3194 JAPAN
kato@sys.im.hiroshima-cu.ac.jp*

²*HIT Laboratory,
University of Washington
Box 352-142, Seattle,
WA 98195, USA
grof@hitl.washington.edu*

³*ATR MIC Laboratories,
ATR International,
2-2 Hikaridai, Seika-cho,
Soraku-gun, Kyoto, Japan
poup@mic.atr.co.jp*

Abstract

In this paper we address the problems of virtual object interaction and user tracking in a table-top Augmented Reality (AR) interface. In this setting there is a need for very accurate tracking and registration techniques and an intuitive and useful interface. This is especially true in AR interfaces for supporting face to face collaboration where users need to be able to easily cooperate with each other. We describe an accurate vision-based tracking method for table-top AR environments and tangible user interface (TUI) techniques based on this method that allow users to manipulate virtual objects in a natural and intuitive manner. Our approach is robust, allowing users to cover some of the tracking markers while still returning camera viewpoint information, overcoming one of the limitations of traditional computer vision based systems. After describing this technique we describe its use in a prototype AR applications.

1. Introduction

In the design session of the future several architects sit around a table examining plans and pictures of a building they are about to construct. Mid-way through the design session they don light-weight see-through head mounted displays (HMDs). Through the displays they can still see each other and their real plans and drawings. However in the midst of the table they can now see a three-dimensional virtual image of their building. This image is exactly aligned over the real world so the architects are free to move around the table and examine it from any viewpoint. Each person has their own viewpoint into the model, just as if they were seeing a real object. Since it is virtual they are also free to interact with the model in real time, adding or deleting parts to the building or scaling portions of it to examine it in greater detail. While interacting with the virtual model they can also see each other and the real world, ensuring a very natural collaboration and flow of communication.

While this may seem to be a far-off vision of the future there are a number of researchers that have already developed table-top AR systems for supporting face-to-face collaboration. In Kiyokawa's work two users are able to collaboratively design virtual scenes in an AR interface and then fly inside those scenes and experience them immersively [Kiyokawa 98]. The AR2 Hockey system of Ohshima et. al. [Ohshima 98] allows two users to play virtual air hockey against each other, while the Shared Space interface supports several users around a table playing a collaborative AR card matching game [Billinghurst 99]. Finally the Emmie system of Butz et. al. [Butz 99] combines virtual three-dimensional AR information with conventional two-dimensional displays in a table-top system that supports face-to-face collaboration.

There are collaborative AR environments that do not rely on a table-top setting, such as Studierstube [Schmalstieg 96], however it is clear that this is an important category of AR interface. This is due to a number of reasons:

- In face-to-face meetings, people typically gather around a table.
- A table provides a location for placing material relative to meeting content.
- A table provides a working surface for content creation.

In creating an AR interface that allows users to manipulate 3D virtual objects in a real table-top there are a number of problems that need to be overcome. From a technical viewpoint we need to consider tracking and registration accuracy, robustness and the overall system configuration

From a usability viewpoint we need to create a natural and intuitive interface and address the problem of allowing real objects to occlude virtual images.

In this paper we describe some computer vision based techniques that can be used to overcome these problems. These techniques have been designed to support a

Tangible Augmented Reality (TAR) approach in which lessons from Tangible User Interface (TUI) design are applied to the design of AR interfaces. In the next section we describe the idea of Tangible AR interfaces in more detail and in section 3 some results from early prototypes of our Table-top AR interfaces. In section 4 our current registration and interaction techniques are described. Finally in section 5 we present our most recent prototype system based on our method and we conclude in section 6.

2. Tangible Augmented Reality

Although there have been many different virtual object manipulation techniques proposed for immersive virtual reality environments, there has been less work conducted on AR interaction techniques. One particularly promising area of research that can be applied is the area of Tangible User Interfaces. The goal of Tangible User Interface research is to turn real objects into input and output devices for computer interfaces [Tangible 2000].

Tangible interfaces are powerful because the physical objects used in them have properties and physical constraints that restrict how they can be manipulated and so are easy to use. However there are limitations as well. It can be difficult to change these physical properties, making it impossible to tell from looking at a physical object what is the state of the digital data associated with that object. In some interfaces there is also often a disconnect between the task space and display space. For example, in the Gorbet's Triangles work, physical triangles are assembled to tell stories, but the visual representations of the stories are shown on a separate monitor distinct from the physical interface [Gorbet 98].

The visual cues conveyed by tangible interfaces are also sparse and may be inadequate for some applications. The ToonTown remote conferencing interface uses real dolls as physical surrogates of remote people [Singer 99]. However the non-verbal and visual cues that these objects can convey is limited compared to what is possible in a traditional videoconference. Showing three-dimensional imagery in a tangible setting can also be problematic because it is dependent on a physical display surface.

Many of these limitations can be overcome through the use of Augmented Reality. We define Tangible Augmented Reality as AR interfaces based upon Tangible User Interface design principles. In these interfaces the intuitiveness of the physical input devices can be combined with the enhanced display possibilities provided by virtual image overlays. Head mounted display (HMD) based AR provides the ability to support independent public and private views of the information space, and has no dependence on physical display

surfaces. Similarly, AR techniques can be used to seamlessly merge the display and task space.

Research in immersive virtual reality point to the performance benefits that can result from a Tangible Augmented Reality approach. The physical properties of the tangible interface can be used to suggest ways in which the attached virtual objects might interact and enhance the virtual interaction. For example, Lindeman finds that physical constraints provided by a real object can significantly improve performance in an immersive virtual manipulation task [Lindeman 99]. Similarly Hoffman finds adding real objects that can be touched to immersive Virtual Environments enhances the feeling of Presence in those environments [Hoffman 98]. While in Poupyrev's virtual tablet work, the presence of a real tablet and pen enable users to easily enter virtual handwritten commands and annotations [Poupyrev 98].

Interfaces that combine Reality and Virtuality are not new. However, Ishii summarizes the state of AR research when he says that AR researchers are primarily concerned with “.. considering purely visual augmentations” rather than the form of the physical objects those visual augmentations are attached to [Ishii 97]. If we are to create more usable AR interfaces then researchers must have a better understanding of design principles based on form as well as function.

In our augmented reality work we advocate designing the form of physical objects in the interface using established Tangible User Interface design methods. Some of the tangible design principles include:

- Object affordances should match the physical constraints of the object to the requirements of the task.
- The ability to support parallel activity where multiple objects or interface elements are being manipulated at once.
- Support for physically based interaction techniques (such as using object proximity or spatial relations).
- The form of objects should encourage and support spatial manipulation
- Support for multi-handed interaction.

Physical interface attributes are particularly important in interfaces designed to support face-to-face collaboration. In this case people commonly use the resources of the physical world to establish a socially shared meaning [Gav 97]. Physical objects support collaboration both by their appearance, the physical affordances they have, their use as semantic representations, their spatial relationships, and their ability to help focus attention. In an AR interface the physical objects can further be enhanced in ways not normally possible such as providing

dynamic information overlay, private and public data display, context sensitive visual appearance, and physically based interactions.

In the next section we describe how the Tangible Augmented Reality approach was applied in an early collaborative table-top AR experience.

3. Case Study: Shared Space Siggraph 99

The Shared Space Siggraph 99 application was designed to explore how augmented reality could be used to enhance face to face collaboration in a table-top setting. In order to do this we aimed to develop a compelling collaborative AR experience that could be used by novices with no training or computer experience. We based this experience on a simple child's card matching game. In our variant three people around a table wear Olympus HMDs with cameras attached (figure 1).

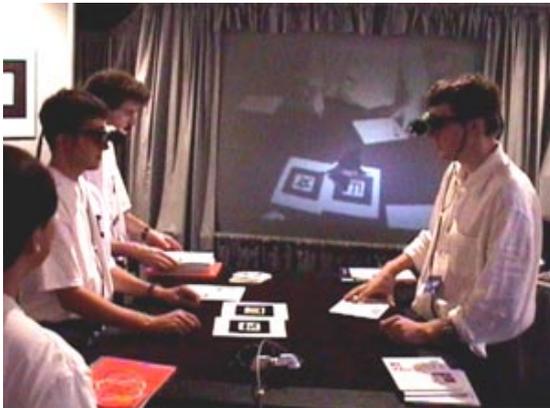


Fig. 1: Users Around the Playing Table

On the table there are large cards with Japanese Kanji characters on them. When the users turn over the cards they see different three-dimensional virtual objects appearing on top of the cards (figure 2).

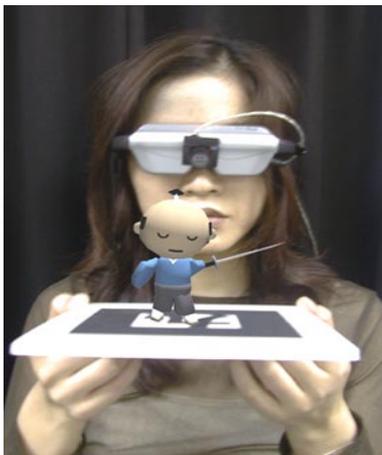


Fig. 2: A Virtual Object on a Card

The goal of the game is to collaboratively match objects that logically belong together. When cards containing correct matches are placed side by side an animation is triggered involving the objects (figure 3a,3b). For example, when the card with the UFO on it is placed next to the card with the alien on it the alien appears to jump into the UFO and start to fly around the Earth. Since the players are all co-located they can easily all see each other and the virtual objects that are being exposed.

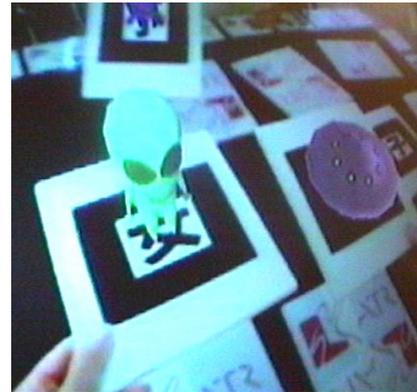


Fig. 3a: Two Matching Objects Being Brought Together

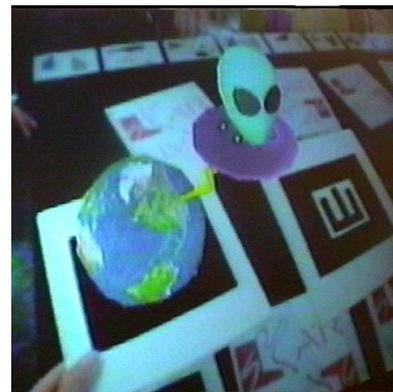


Fig. 3b: The Virtual Object Interaction

The HMD and camera are connected to an SGI O2 computer that performs image processing on the video input and composites computer graphics onto the image for display in the HMD. The users experience a video see-through augmented reality, seeing the real world through the video camera. The real cards are all labeled with square tracking markers. When users look at these cards, computer vision techniques are used to find the tracking mark and determine the exact pose of the head mounted camera relative to it [Kato 99a]. Once the position of the real camera is known, a virtual image can then be exactly overlaid on the card. Figure 4 overleaf summarizes the tracking process.

Although this is a very simple application it provides a good test of the usefulness of the tangible interface metaphor for manipulating virtual models. The Kanji

characters are used as tracking symbols by the computer vision software and were mounted on flat cards to mimic the physical attributes people were familiar with in normal card games. This was to encourage people to manipulate them the same way they would use normal playing cards. However, the tracking patterns needed to be placed in such a way that people would not cover them with their hands when picking the cards up, and they needed to be large enough to be seen from across the table. So there was a design trade-off between making the cards large enough to be useful for the tracking software and too large that they could not easily be handled. The physically based interaction techniques were also chosen based on natural actions people perform with playing cards, such as turning them over, rotating them, holding them in the hands, passing them to each other and placing them next to each other.

3.1 User Experiences

The Shared Space demonstration has been shown at the SIGGRAPH 99 and Imagina 2000 conferences and the Heniz-Nixdorf museum in Germany. Over 3,500 people have tried the software and given us feedback.

Users had no difficulty with the interface. They found it natural to pick up and manipulate the physical cards to view the virtual objects from every angle. Once they held a card in view and could see a virtual object, players typically only made small head motions. However it was common to see people rotating the cards at all angles to see the virtual objects from different viewpoints. Since the matches were not obvious some users needed help from other collaborators at the table and players would often spontaneously collaborate with strangers who had the matching card they needed. They would pass cards between each other, and collaboratively view objects and completed animations. They almost always expressed surprise and enjoyment when they matched virtual objects and we found that even young children could play and enjoy the game. Users did not need to learn any complicated computer interface or command set. The only instructions people needed to be given to play the game was to turn the cards over, not cover the tracking

patterns and to find objects that matched each other.

At the Imagina 2000 conference 157 people filled out a short user survey. They were asked to answer the following questions on a scale of one to seven (*1= very easily/real* and *7 = not very easily/real*):

- 1: How easily could you play with other people ?
- 2: How real did the virtual objects seem to you?
- 3: How easily could you interact with the virtual objects?

Table 1 summarizes the results. As can be seen, users felt that they could very easily play with the other people (5.64) and interact with the virtual objects (5.62). Both of these are significantly higher than the neutral value of 3.5; the t-test value row showing the results from a one-tailed t-test. It is also interesting that even though the virtual object were not real, on average people rated them as being midway between not very real and very real. When asked to fill what they enjoyed most about the system the top three responses were: the interactivity (25), the ease of use (18), and how fun it was (15).

	Q1 (n=132)	Q2 (n=157)	Q3 (n=157)
Average	5.64	4.01	5.62
Std Dev.	1.19	1.20	1.20
t-test val.	237.09	66.70	278.74

Table 1: Shared Space Survey Results

These results illustrate that by applying a tangible interface metaphor we are very able to create a compelling table-top AR experience in which the technology was transparent. In the next section we describe in more detail our current tracking and interaction techniques which overcome some of the limitations of the Shared Space Siggraph 99 application, including occlusion of virtual images by real objects, robust tracking, and a limited range of tangible interaction methods.

4. An Improved Method

In the previous section we described our Shared Space

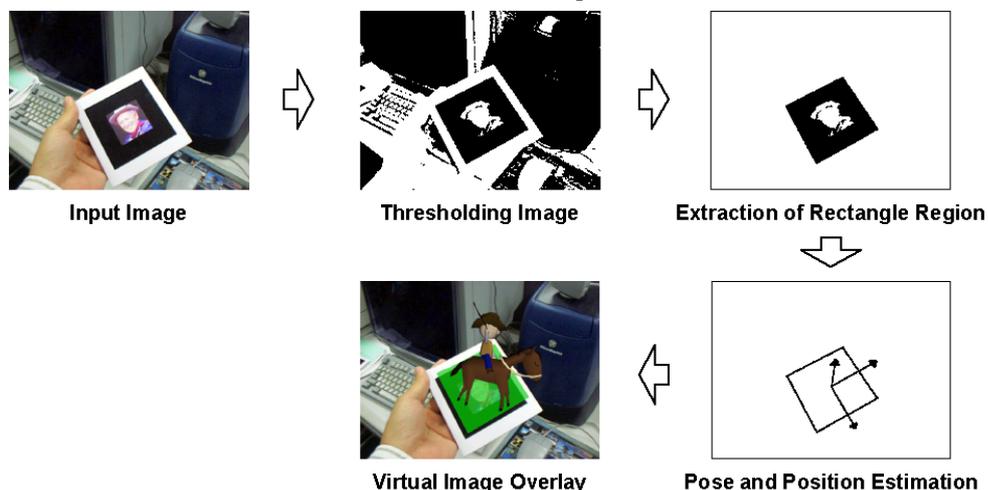


Figure 4: The Vision-Based AR Tracking Process

Siggraph 99 collaborative AR application which was based on our computer vision tracking technique and a TUI design method. Although users found this a successful Tangible AR interface and were able to collaborate easily with each other, there were a number of shortcomings. First the tracking method only provided user head position relative to each of the cards in view, not to any global world coordinate system. This makes it difficult to implement certain types of Tangible Interaction techniques. Secondly, since the vision-based tracking used single large markers the system failed when a tracking marker was partially covered by a user's hand or other object. Finally, we didn't solve the problem of the real cards not being able to occlude the virtual models on other cards, causing foreground/background confusion. In this section we describe a new approach to table-top AR that overcomes these limitations.

4.1 Implementing Global Coordinate Tracking

In order to track user and object position we modified the table-top AR environment by attaching tracking fiducials to the table top surface. Figure 5 shows the new system configuration.

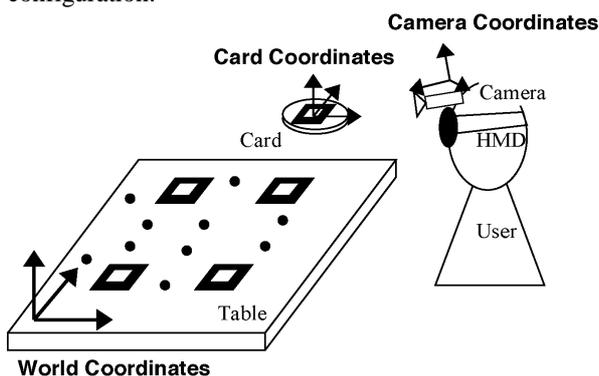


Figure 5 Table-top Configuration.

The table-top fiducials consist of a mixture of square tracking patterns with small circular blobs between them. We define the world coordinates frame as a set of coordinate axes aligned with the table surface. The camera attached to the HMD detects the self-pose and position in the world coordinates by looking at multiple fiducials on the table. In section 4.2 we describe the vision-based tracking method used for head tracking from multiple fiducials. Our method is robust to partial occlusion, so users can move their hands across the table-top and the camera position is still reliably tracked. Finding the user head position in world coordinates means that 3D virtual objects can also be represented in the world coordinates and the user can see them appearing on the on the real table.

The user can also still pick up an object on which a fiducial is drawn, and our previous method can be used to

calculate the relationship between the object and camera coordinates. However because the camera pose in world coordinates is known, we can now find the object pose in the world coordinate frame. Using this information we can use new manipulation methods based on object pose and movement. These are described in section 4.4.

Since this configuration uses only one camera as a sensor, it is compact and could be portable. Even if there are multiple people around the table, the systems for each user do not interfere so our global tracking approach scales to any number of users. In fact, information from several users could be integrated to increase the accuracy or robustness, although this still needs to be done.

4.2 Tracking of Multiple Fiducials

Our previous tracking method provides satisfactory accuracy for a table-top AR environment, however it uses a single relatively large square marker as a fiducial. So if a hand or other object to even partially overlapped the fiducial the tracking was lost. This decreased the robustness of tracking under the conditions where a hand could overlap the fiducials. Also if there is some distance between tracked fiducials and displayed virtual objects, tracking errors strongly influence the registration accuracy. That is, using a single fiducial decreases the accuracy of registration under the conditions where virtual objects need to be displayed around on the table.

We have developed a new tracking method in which multiple large square and blobs are used as fiducials and pose and position are estimated from all of the detected fiducial marks. This means that many of the fiducial can be covered up without losing tracking. Many tracking methods using multiple markers have been proposed at such conferences as IWAR99 or ISMR99. However there are few methods that use combination of different types of tracking markers.

The square marker used previously has the characteristic that 3D pose and position can be estimated from a single marker. The same results can be achieved by using a set of circular blobs. Since circular blobs are relatively small and can be spread over a wider area, it is more difficult to cover them all. However the disadvantage is that three blobs are required for pose and position estimation and identification of each blob is difficult from visible features. Therefore another method for identification of each blob has to be adopted. Our tracking method uses the features of both the square and blob markers. As shown in figure 6, multiple squares and blobs lie on the table spread over a wide area. The relationships among all markers are known and are described in world coordinates.

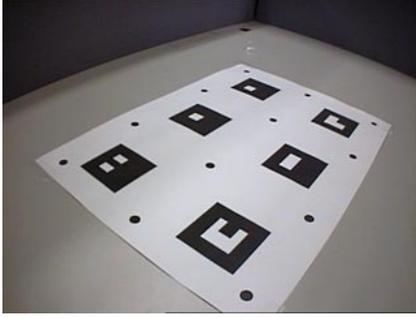


Figure 6 An Example of Fiducials.

Considering just the square markers, there are two situations that might occur in the captured video image:

- 1) One or more square markers are visible.
- 2) No square markers are visible.

In the rest of this section we explain how we can achieve robust pose tracking in each of these circumstances.

1) One or More Squares are Visible

If there is a square marker in the image, it is possible to estimate 3D pose and position using our earlier method [Kato 99a]. However if there is more than one square visible we can achieve more robust tracking if we estimate pose from all of available features. In order to do this we adopt following procedures:

step 1) The biggest square marker is selected in the image. 3D pose and position are initially estimated from it using our earlier method. This information is represented as the following transformation function from marker coordinates to camera coordinates:

$$(x_c, y_c, z_c) = \text{trans}(x_w, y_w, z_w) \quad (\text{eq.1})$$

where (x_w, y_w, z_w) is a position in world coordinates and (x_c, y_c, z_c) is the same position in camera coordinates.

step 2) The positions of all the circular blobs are estimated in screen coordinates by using the above transformation function, a projective function and the 3D positions of blobs in the world coordinates:

$$(x_s, y_s) = \text{perspect}(\text{trans}(x_w, y_w, z_w)) \quad (\text{eq.2})$$

where the function *perspect* is a projective function. This function consists of perspective projection parameters and image distortion parameters [Kato 99b].

step 3) The actual screen coordinates of the detected blobs are compared to the estimated positions. Using the positions of all successfully matched blob markers and the 4 vertices of all extracted square markers, the 3D pose and position are re-estimated. For this calculation, the initial transformation

function is used and modified as the amount of errors between the actual feature positions in the image and the estimated positions goes to minimum using a hill-climbing method.

2) No Square Markers are Visible

In this case, we assume that some of the circular blobs are visible so a procedure for robust identification of blob markers is needed. If we assume that the video capture rate is sufficiently fast then there is little difference in blob position between frames. So we can use the blobs positions that are estimated at last frame containing a square marker and then track these over subsequent frame. The blob positions in the frame with the square marker are found using the above method.

This method of tracking blobs from frame to frame works well when head motion is not too fast and a hand moves to overlap some of the square markers. As we discovered in the Shared Space Siggraph 99 application, rapid hand motion is more likely than rapid head motion. However if the head moves quickly in condition where only dot markers can be seen the tracking will fail. In order to decrease this possibility the layout of fiducials is also important.

Figure 7 shows an example of the tracking. In figure 7a both square and blob markers are visible, while in figure 7b some square markers are covered by a hand. In this case, we can see that virtual objects are still displayed on the correct position. However, we can also see the incorrect occlusion between the virtual objects and the hand. In the next section we describe how to address this problem.

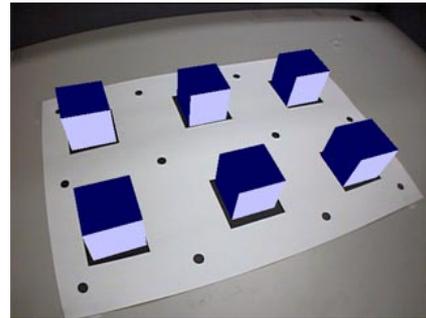


Figure 7a: Virtual Objects on Multiple Markers

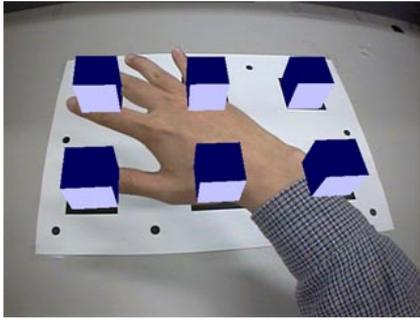


Figure 7b: Markers Covered by a Hand

4.3 The Occlusion Problem

When integrating real and virtual objects, if depth information is not available, problems with incorrect occlusion can result. That is, a virtual object which should be far from the user sometimes occludes a real object that is nearer to the user. This problem prevents a user from recognizing depth information and decreases usability. Yokoya proposed a method that overcomes this problem by getting depth information from stereo cameras [Yokoya 99]. This could be achieved by two cameras and fast computer.

With regard to table-top virtual object manipulation this problem mostly arises between a hand which manipulates virtual objects and the virtual objects on the table. As the person moves their hand above the table the virtual objects on the table surface incorrectly appear in front of the hand (see figure7b). Considering this problem we arrived at the following solutions.

- 1) We restrict users to interacting with virtual images with physical objects they hold in their hands. These objects can have a fiducial marker on them so the position and pose can be detected. Also the shape of the object is known. Thus using virtual models of the hand-held real objects we can correctly occlude the virtual models. That is, far-off virtual objects might cover the user's hand but the real object manipulating the virtual objects correctly occludes them. We hypothesize that this will affect usability less than a total absence of occlusion support.
- 2) Since there are no virtual objects in the naturally occurring in the real world, we think that user's will not find it unnatural that virtual objects have transparency. Therefore we hypothesize that a user will not object if virtual objects cannot completely occlude real objects. This is especially the case in optical-see through AR where every virtual object is at least a little transparent making it is difficult for them to cover a real object perfectly.

These can be realized by using Alpha-buffer and Z-buffer

information when rendering. Figure 8a shows a physical object correctly occluding virtual objects. In this figure, we can see all depth information is correctly represented except for the hand.

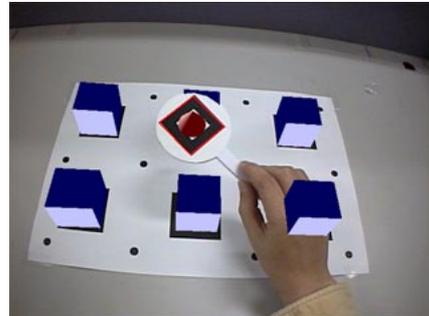


Figure 8a: correct overlay of a physical object

Figure 8b shows virtual objects with a little transparency. In this case, even if the depth information of the hand is still incorrect, we can see the hand because of the transparency, reducing the visual discrepancy.

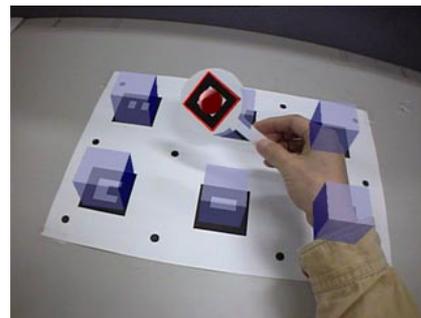


Figure 8b: transparent virtual objects

4.4 Implementing Natural and Intuitive Manipulation

In the Shared Space Siggraph 99 application users were able to easily interact with the application because the physically based interaction techniques matched the affordances of the real cards. However because the cards were not tracked relative to global coordinates there were only a limited number of manipulation methods that could be implemented.

If the virtual objects are attached to a card, or manipulated by a card there are a number of other possible manipulation methods that could be explored:

- *Inclining:* If the card the virtual object is on is tilted, the object should slide across the card surface.
- *Pushing down:* When a card pushes down a virtual object on the table, it should disappear into the table.
- *Picking & pulling:* When a card picks a virtual object on the table from above it, it should

appear to be connected with a card by short virtual string. Pulling the string can then move it.

- *Shaking*: When shaking a card, an object could appear on the card or change to another object.

Some of these commands simulate physical phenomena in the real world and other simulate table magic. In all these cases we establish a cause-and-effect relationship between physical manipulation of the tangible interface object and the behavior of the virtual images.

These behaviors can be implemented using knowledge about the real object position and orientation in world coordinates. There are two classes of physical interaction techniques. One in which behaviors can be determined purely from knowing the relationship between card coordinates and camera coordinates. Card *shaking* belongs to this class. The other is a class in which behaviors can be determined by using two relationships: between card and camera coordinates and between world and camera coordinates. Behaviors such as *inclining*, *picking* and *pushing* belong to this class. In the remainder of this section we show how to recognize examples of these behaviors.

Detecting Type A Behaviors: *Shaking*

A series of detected transformation matrices from the card to camera coordinate frames are stored over time. Observing rotation and translation components from these matrices, the user behavior can be determined. For the *shaking* behavior,

- 1) The pose and position at $t[\text{sec}]$ before the current time are almost same as current pose and position.
- 2) There is little changes in the card rotation period.
- 3) There is a time when the card is moved farther than y [mm] in surface plane of the card.
- 4) There is little movement in the surface normal direction of the card.

When all the above conditions are satisfied, it is assumed that the user is *shaking* the physical card and the corresponding *shaking* command is executed.

Detecting Type B Behaviors: *Inclining and Pushing*

When the camera pose and position and a card pose and position are detected, a transformation matrix between the card coordinate frame and world coordinate frame can be calculated. Observing the rotation and translation components of this transformation matrix, behaviors such as card tilting and pushing can be determined. At this time, the pose, position and size of virtual objects on the table are also be used to determine the user interaction.

5. Prototype System

We are currently developing a prototype table-top AR

system for virtual interior design using the interaction and tracking techniques described above. Figure 9 shows the current version of this prototype. As can be seen users are able to user a real paddle to move around virtual objects in the AR interface. There is correct occlusion between the paddle and the virtual objects and transparency cues are use to minimize the hand occlusion problem. Multiple users can gather around the table-top and simultaneously interact with the virtual scene. Using this system, we plan to conduct user studies to explore the effects of Tangible AR interfaces on face to face collaboration.

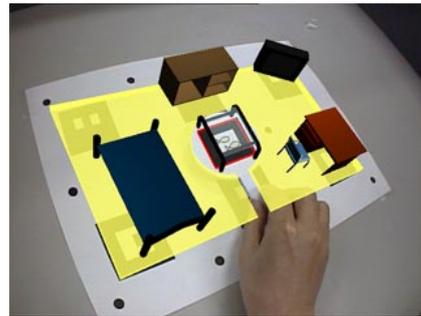


Figure 9 A Prototype of an Interior Design Application

6. Conclusions

In this paper we addressed the problems of virtual object interaction and user tracking in a table-top Augmented Reality (AR) interface. We first described an approach to AR interface design based on Tangible User Interface design principles. Next we showed how using these design principles we were able to create a compelling table-top AR experience which could be used by novices with no computer experience. Coupling a tangible interface with AR imagery achieved a technology transparency that enhanced face to face collaboration. However there were problems with the tracking approach and the limited types of interaction method support in the Shared Space Siggraph 99 experience.

In the second half of the paper we address these issues. We presented a more accurate and robust vision-based tracking method for table-top AR environments that finds pose information from multiple fiducial marks. This tracking technique also allows us to track users and card in world coordinates. Tangible user interface (TUI) techniques based on this method that allow users to manipulate virtual objects in a natural and intuitive manner. We are currently developing a virtual interior design application so we can further explore the effect of AR tangible user interface in table-top collaboration.

References

[Billingham 99] Billingham, M., Kato, H., Kraus, E., May, R. Shared Space: Collaborative Augmented Reality.

In *Visual Proceedings, SIGGRAPH 99*, August 7-12th, Los Angeles, CA, ACM Press, 1999.

[Butz 99] A. Butz, T. Höllerer, S. Feiner, B. MacIntyre, C. Beshers, Enveloping Users and Computers in a Collaborative 3D Augmented Reality, In *Proc. IWAR '99*, San Francisco, CA, October 20-21, 1999, pp. 35-44.

[Gav 97] Gav, G., Lentini, M. Use of Communication Resources in a Networked Collaborative Design Environment.

http://www.osu.edu/units/jcmc/IMG_JCMC/ResourceUse.html

[Gorbet 98] Gorbet, M., Orth, M., Ishii, H. Triangles: Tangible Interface for Manipulation and Exploration of Digital Information Topography. In *Proceedings of CHI 98*, Los Angeles, CA, 1998.

[Hoffman 98] Hoffman, H. Physically Touching Virtual Objects Using Tactile Augmentation Enhances the Realism of Virtual Environments. In *Proceedings of Virtual Reality Annual International Symposium (VRAIS '98)*, 1998, pp. 59-63.

[Ishii 97] Ishii, H., Ullmer, B. Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms. In *Proceedings of CHI 97*, Atlanta, Georgia, USA, ACM Press, 1997, pp. 234-241.

[Kato 99a] H. Kato, M. Billinghurst: Marker Tracking and HMD Calibration for a Video-based Augmented Reality Conferencing System, In *Proc. IWAR '99*, San Francisco, CA, October 20-21, 1999, pp.85-94.

[Kato 99b] H. Kato, M. Billinghurst, K. Asano, K. Tachibana, An Augmented Reality System and its Calibration based on Marker Tracking, *Transactions of the Virtual Reality Society of Japan*, Vol.4, No.4, pp.607-616, 1999 (in Japanese).

[Kiyokawa 98a] Kiyokawa, K., Iwasa, H., Takemura, H., Yokoya, N. Collaborative Immersive Workspace through a Shared Augmented Environment, In *Proceedings of the International Society for Optical Engineering '98 (SPIE '98)*, Vol.3517, pp.2-13, Boston, 1998.

[Lindeman 99] Lindeman, R., Sibert, J., Hahn, J. Towards Usable VR: An Empirical Study of User Interfaces for Immersive Virtual Environments. In *Proceedings of CHI 99*, 15th-20th May, Pittsburgh, PA, 1999, pp. 64-71.

[Ohshima 98] Ohshima, T., Sato, K., Yamamoto, H., Tamura, H. AR²Hockey: A case study of collaborative augmented reality, In *Proceedings of VRAIS'98*, 1998, IEEE Press: Los Alamitos, pp.268-295.

[Poupyrev 98] Poupyrev, I., Tomokazu, N., Weghorst, S., Virtual Notepad: Handwriting in Immersive VR. In *Proceedings of IEEE VRAIS'98*, 1998, pp.126-132.

[Schmalsteig 96] Schmalsteig, D., Fuhrmann, A., Szalavari, Z., Gervautz, M., Studierstube - An Environment for Collaboration in Augmented Reality. In *CVE '96 Workshop Proceedings*, 19-20th September 1996, Nottingham, Great Britain.

[Singer 99] Singer, A., Hindus, D., Stifelman, L., White, S. Tangible Progress: Less is More in Somewire Audio Spaces. In *Proceedings of CHI 99*, 15th-20th May, Pittsburgh, PA, 1999, pp. 104 – 111.

[Tangible 2000] MIT Media Lab, Tangible Media Group <http://tangible.www.media.mit.edu/groups/tangible/>

[Yokoya 99] N. Yokoya, H. Takemura, T. Okuma, M. Kanbara, Stereo Vision Based Video See-through Mixed Reality, *Mixed Reality (Proc. Of ISMR99)*, Springer-Verlag, 1999, pp.131-145.

Usability Evaluation in Virtual Environments: A Comparison and Integration of Methods

Doug A. Bowman, Joseph L. Gabbard, and Deborah Hix¹

Categories and Subject Descriptors: H.5.2 [Information Interfaces and Presentation]: User Interfaces – Evaluation/methodology; Theory and methods

General Terms: Design, Experimentation, Human factors, Performance

Additional Keywords and Phrases: Usability evaluation, Virtual environments, Three-dimensional user interfaces, Multi-modal user interfaces

Abstract

Virtual environments (VEs) are a relatively new type of human-computer interface in which users perceive and act in a three-dimensional world. The designers of such systems cannot rely solely on design guidelines for traditional two-dimensional interfaces, so usability evaluation is crucial for VEs. We present an overview of VE usability evaluation. First, we present some of the issues that differentiate VE usability evaluation from evaluation of traditional user interfaces such as GUIs. We also present a review of VE evaluation methods currently in use, and discuss a simple classification space for VE usability evaluation methods. This classification space provides a structured means for comparing evaluation methods according to three key characteristics: involvement of representative users, context of evaluation, and types of results produced. To illustrate these concepts, we compare two existing evaluation approaches: testbed evaluation [Bowman, Johnson, & Hodges, 1999], and sequential evaluation [Gabbard, Hix, & Swan, 1999]. We conclude by presenting a novel integrated approach to VE usability evaluation that employs both of these approaches.

1 Introduction and motivation

During the past several years, virtual environments (VEs) have gained broad attention throughout the computing community. During roughly that same time period, usability has become a major focus of interactive system development. *Usability* can be broadly defined as “ease of use” plus “usefulness”, including such quantifiable characteristics as learnability, speed of user task performance, user error rate, and subjective user satisfaction [Hix & Hartson 1993; Shneiderman 1992]. Despite intense and widespread research in both VEs and usability, until recently there were very few examples of research coupling VE technology with usability — a necessary coupling if VEs are to reach their full potential. By focusing on usability from the very beginning of the development process, developers are more likely to avoid creating interaction techniques (ITs) that do not match appropriate user tasks or producing standards and principles for VE user interface development that are nonsensical. In this paper, we focus on *usability evaluation* of VEs — determining how different ITs, interface styles, and numerous other factors such as information organization, visualization, and navigation affect the usability of interactive tasks in VEs.

Although numerous methods exist for usability evaluation of interactive computer applications, these methods have well-known limitations, especially for evaluating VEs. For example, most usability evaluation methods are applicable only to a narrow range of interface types (e.g., graphical user interfaces, or GUIs) and have had little or no use with innovative, non-routine interfaces such as those found in VEs. VE applications have interaction styles so radically different from ordinary user interfaces that well-proven methods that produce usable GUIs may be neither appropriate nor effective. The focus of most existing methods, while properly user-task-based, is on a single user performing isolated, low-level user tasks —

¹ Department of Computer Science, Virginia Tech, Blacksburg, Virginia 24061, {bowman, jgabbard, hix}@vt.edu

very different than the typical VE in which one or more users are performing integrated, shared, multi-threaded tasks.

There have been attempts to adapt traditional usability evaluation methods for use in VEs, and a few notable efforts to develop structured usability evaluation methods for VEs. In this paper, we present a survey of existing approaches to usability evaluation to VEs. We begin by making explicit some of the important differences between evaluation of VE user interfaces and traditional GUI evaluation. Next, we categorize usability evaluation techniques into a space based on three important characteristics. Two major approaches are presented and compared: *testbed evaluation*, which focuses on low-level ITs in a generic context, and *sequential evaluation*, which applies several different types of evaluation methods within the context of a particular VE application. Finally, we present an integration of these two approaches that provides more power and an even broader set of techniques to the VE developer or researcher.

We would like to set the context for this paper by explaining some terminology. First, we take a broad approach to assessing *usability*: it includes any characteristic relating to the ease of use and usefulness of an interactive software application, including user task performance, subjective satisfaction, user comfort, and so on. We define *usability evaluation* as assessment of a specific application's user interface (often at the prototype stage), an interaction metaphor or technique, or an input device, for the purpose of determining its actual or probable usability. *Usability engineering* is in general a term covering the entire spectrum of user interaction development activities, including user and task analysis, conceptual and detailed user interaction design, prototyping, and numerous methods of usability evaluation. The roles involved in usability evaluation include a *developer* (who implements the application and/or interface software), an *evaluator* (who conducts evaluation sessions – may be the same person as the developer), and a *user* or *subject* (who participates in evaluation sessions).

2 Distinctive characteristics of VE evaluation

The approaches we present for usability evaluation of virtual environments have been developed and used in response to perceived differences between the evaluation of VEs and the evaluation of traditional user interfaces such as GUIs. Many of the fundamental concepts and goals are similar, but use of these approaches in the context of VEs is distinct. Here, we present some of the issues that differentiate VE usability evaluation, organized into several categories. The categories contain overlapping considerations, but provide a rough partitioning of these important issues.

2.1 Physical environment issues

One of the most obvious differences between VEs and traditional interfaces is the *physical environment* in which the interface is used. In VEs, non-traditional input and output devices are used, which can preclude the use of some types of evaluation. Users may be standing rather than sitting, and they may be moving about a large space, using whole-body movements. These properties give rise to several issues for usability evaluation. Following are some examples:

- In interfaces using non-see-through head-mounted displays (HMDs), the user cannot see the surrounding physical world. Therefore, the evaluator must ensure that the user will not bump into walls or other physical objects, trip over cables, or move outside the range of the tracking device. A related problem in surround-screen VEs (such as the CAVE™) is that the physical walls can be difficult to see because of projected graphics. Problems of this sort could contaminate the results of a usability evaluation (e.g., if the user trips while in the midst of a timed task), and more importantly could cause injury to the user. To deal with this, the evaluator can ensure that cables are bundled and will not get in the way of the user (e.g., cables may descend from above). Also, the user may be placed in a physical enclosure that limits movement to areas where there are no physical objects to interfere.
- Many VE displays do not allow multiple simultaneous viewers (e.g., user and evaluator), so equipment must be set up so that an evaluator can see the same image as the user. With an HMD, for example, this can be done by splitting the video signal and sending it to both the HMD and a monitor. In a surround-screen or tabletop VE, a monoscopic view of the scene could be rendered

to a monitor, or, if performance will not be adversely affected, both the user and the evaluator can be tracked (this can cause other problems, however – see section 2.2 on evaluator considerations). If images are viewed on a monitor, then it is difficult to see both the actions of the user and the graphical environment at the same time, meaning that multiple evaluators may be necessary to observe and collect data during an evaluation session.

- A common and very effective technique for generating important qualitative data during usability evaluation sessions is the “think aloud” protocol. With this technique, subjects talk about their actions, goals, and thoughts regarding the interface while they are using the application. In some VEs, however, voice recognition is used as an IT, rendering the think aloud protocol much more difficult and perhaps even impossible. Post-session interviews may help to recover some of the information that would have been obtained from the think aloud protocol.
- Another common technique involves recording video of both the user and the interface. Since VE users are often mobile, a single, fixed camera may require a very wide shot, which may not allow precise identification of actions. This could be addressed by using a tracking camera (additional expense and complexity) or a camera operator (additional personnel). Moreover, views of the user and the graphical environment must be synchronized so that cause and effect can clearly be seen on the videotape. Finally, the problem of recording video of a stereo graphics image must be overcome.
- An ever-larger number of proposed VE applications are shared among two or more users. These collaborative VEs become even more difficult to evaluate than single-user VEs because of physical separation between users (i.e., different users are in more than one physical location), the additional information that must be recorded for each user, the unpredictability of network behavior as a factor influencing usability, the possibility that each user will have different input and output devices, and the additional complexity of the system, which may cause more frequent crashes or other problems.

2.2 Evaluator issues

A second set of issues relates to the role of the evaluator in a VE usability evaluation. Because of complexities and distinctive characteristics of VEs, a usability study may require multiple evaluators, different evaluator behaviors, or both. Following are some examples:

- Many VEs attempt to produce a sense of *presence* in the user; that is, a feeling of actually being in the virtual world rather than the physical one. Evaluators can cause breaks in presence if they can be sensed by the user. In VEs using projected graphics, the user will see an evaluator if the evaluator moves into the user’s field of view. This may break presence since the evaluator is not part of the virtual world. In any type of VE, touching or talking to the user can cause such breaks. If the evaluation is measuring presence, or if presence is hypothesized to affect performance on the task being evaluated, then the evaluator must take care to remain unsensed during the evaluation.
- Because breaks in presence are so important, an evaluator probably does not wish to intervene at all during an evaluation session. This means that the experimental application/interface must be robust and bug-free, so that the session does not have to be interrupted to fix a problem. Also, instructions given to the user must be very detailed, explicit, and precise, and the evaluator should make sure the user has a complete understanding of the procedure and tasks before beginning the session.
- VE hardware and software are often more complex and less robust than traditional user interface hardware and software. Again, multiple evaluators may be needed to do tasks such as helping the user with display and input hardware, running the software that produces graphics and other output, recording data such as timings and errors, and recording critical incidents and other qualitative observations of a user’s actions.
- Traditional user interfaces typically require only a discrete, single stream of input (e.g., from mouse and keyboard), but many VEs include multi-modal input, combining discrete events,

gestures, voice, and/or whole-body motion. It is much more difficult for an evaluator to process these multiple input streams simultaneously and record an accurate log of the user's actions. This makes multiple evaluators and video even more important.

2.3 User issues

There are also a large number of issues related to the user population used as subjects in VE usability evaluations. In traditional evaluations, subjects are gleaned from the target user population of an application or from a similar representative group of people. Efforts are often made, for example, to preserve gender equity, to have a good distribution of ages, and to test both experts and novices, if these differences are representative of the target user population. The nature of VE evaluation, however, does not always allow for such straightforward selection of users. Following are some examples:

- VEs are still often a “solution looking for a problem.” Because of this, the target user population for a VE application or IT to be evaluated may not be known or well-understood. For example, a study comparing two virtual travel techniques is not aimed at a particular set of users. Thus, it may be difficult to generalize performance results. The best course of action is to evaluate the most diverse user population possible in terms of age, gender, technical ability, physical characteristics, and so on, and to include these factors in any models of performance.
- It may be impossible to differentiate between novice and expert users, since there are very few potential subjects who could be considered experts in VEs. Most users who could be considered experts might be, for example, research staff, whose participation in an evaluation could confound the results. Also, because most users are typically novices, the evaluation itself may need to be at a lower cognitive and physical level. Evaluators can make no assumptions about a novice user's ability to understand or use a given IT or device.
- Because VEs will be novel to many potential subjects, the results of an evaluation usually exhibit high variability and differences among individuals. This means that the number of subjects needed to obtain a good picture of performance may be higher than for traditional usability evaluations. If statistically significant results are required (depending on the type of usability evaluation being performed), the number of subjects may be even greater.
- Researchers are still studying a large design space for VE ITs and devices. Because of this, evaluations often compare two or more techniques, devices, or combinations of the two. To perform such evaluations using a within-subjects design, users must be able to adapt to a wide variety of situations. If a between-subjects design is used, a larger number of subjects will again be needed.
- VE evaluations must consider the effects of simulator sickness and fatigue on subjects. There are still no definitive results on the causes of simulator sickness, or the acceptable exposure time to VEs, so a worst-case assumption must be made. A lengthy experiment (anything over 30 minutes might be considered lengthy) must contain planned rest breaks and contingency plans in case of ill or fatigued subjects. Shortening the experiment is often not an option, especially if statistically significant results are needed.
- Because we do not know exactly what VE situations cause sickness or fatigue, most VE evaluations should include some measurement (e.g., subjective, questionnaire-based [e.g., Kennedy et al, 1993], or physiological) of these factors. A result indicating that IT X was 50 percent faster than any other evaluated technique would be severely misleading if technique X also made 30 percent of subjects sick! Thus, user comfort measurements should be included in low-level VE evaluations.
- Presence is another example of a measure often required in VE evaluations that has no analogue in traditional user interface evaluation. VE evaluations must often take into account subjective reports of perceived presence, perceived fidelity of the virtual world, and so on. Questionnaires [e.g., Witmer & Singer, 1998] have been developed that purportedly obtain reliable and consistent measurements of such factors.

2.4 Issues related to type of usability evaluation

Traditional usability evaluation can take many forms. These include informal user studies, formal experiments, task-based usability studies, heuristic evaluations, and the use of predictive models of performance (see section 3 for further discussion of these types of evaluations). There are several issues related to the use of various types of usability evaluation in VEs. Following are some examples:

- Evaluations based solely on heuristics (i.e., design guidelines), performed by usability experts, are very difficult in VEs because of a lack of published, verified guidelines for VE user interface design. There are some notable exceptions [Bowman, 2001; Gabbard, 1997; Kaur, 1998], but for the most part it is difficult to predict the usability of a VE interface without studying real users attempting tasks in the VE. It is not likely that a large number of heuristics will appear at least until input and output devices used for VEs are more standardized. Even assuming standardized devices, however, the design space for VE ITs and interfaces is very large, making it difficult to produce effective heuristics to use as the basis for evaluation.
- Another major type of usability evaluation that does not employ users is the application of performance models (e.g., GOMS, Fitts' Law). Again, such models simply do not exist at this stage of VE development. However, the lower cost of both heuristic evaluation and performance model application makes them very attractive for evaluation, so work in these areas will be important.
- Because of the complexity and novelty of VEs, the applicability or utility of automated, tool-based evaluation may be greater than it is for more traditional user interfaces. For example, several issues above have noted the need for more than one evaluator in a VE usability evaluation session. Automated usability evaluations could reduce the need for several evaluators in a single session. There are at least two possibilities for automated usability evaluation of VE user interfaces: first, to automatically collect and/or analyze data generated by one or more users in a VE, and second, to perform an analysis of an interface design using an interactive tool that embodies design guidelines (similar to heuristics). Some work has been done on automatic collection and analysis of data using specific types of repeating patterns in users' data as indicators of potential usability problems (e.g., [Siochi & Hix, 1991]). However this work was performed on a typical GUI, and there appears to be no research yet conducted that studies automated data collection and evaluation of users' data in VEs. Thus, differences in use of these kinds of data for VE usability evaluation have not been explored, but they would involve, at a minimum, collating data from multiple users in a single session, possibly at different physical locations and even in different parts of the VE. At least one tool, MAUVE (Multi-Attribute Usability evaluation tool for Virtual Environments), is being developed [Stanney, personal communication]. MAUVE incorporates design guidelines organized around several VE categories such as navigation, object manipulation, input, output (e.g., visual, auditory, haptic), and so on. Within each of these categories, MAUVE presents a series of questions to an evaluator, who uses the tool to perform a multi-criteria heuristic-style evaluation of a specific VE user interface. Further work in both of these types of automated usability evaluation is of interest, especially in light of the expense of developing and evaluating VEs.
- When performing formal experiments to quantify and compare the usability of various VE ITs, input devices, interface elements, and so on, it is often difficult to know which factors have a potential impact on the results. Besides the primary independent variable (e.g., a specific IT), there are a large number of other potential factors that could be included, such as environment, task, system, or user characteristics. One approach is to try to vary as many of these potentially important factors as possible during a single experiment. Such "testbed evaluation" [Bowman, Johnson, & Hodges, 1999] (see Section 3.2) has been done with some success. The other extreme would be to simply hold as many of these other factors as possible constant, and evaluate only in a particular set of circumstances. Thus, formal VE experimental evaluations may be either overly simplistic or overly complex – finding the proper balance is difficult.

2.5 Miscellaneous issues

- VE usability evaluations generally focus at a lower level than traditional user interface evaluations. In the context of GUIs, a standard look and feel and a standard set of interface elements and ITs exist, so evaluation usually looks at subtle interface nuances or overall interface metaphors. In the VE field, however, there are no interface standards, and we do not have a good understanding of the usability of various interface types. Therefore, VE evaluations most often compare lower-level components, such as ITs or input devices.
- It is tempting to over-generalize the results of evaluations of VE interaction performed in a generic (non-application) context. However, because of the fast-changing and complex nature of VEs, one cannot assume anything (display type, input devices, graphics processing power, tracker accuracy, etc.) about the characteristics of a real VE application. Everything has the potential to change. Therefore, it is important to include information about the environment in which the evaluation was performed, and to evaluate in a range of environments (e.g., using different devices) if possible.

3 Current evaluation methods

A review of recent VE literature indicates that a growing number of researchers and developers are considering usability at some level. Some are employing extensive usability evaluation techniques requiring a representative user base (e.g., [Hix et al, 1999]), while others undertake a less labor-intensive effort such as review and inspection by a usability expert (e.g., [Steed & Tromp, 1998]). While it is clear that VE designers have a number of usability evaluation methods to choose from, it is not yet clear which methods are most cost-effective given a specific development or research scenario.

From the literature, we have compiled a list of usability evaluation methods that have been successfully applied to VEs. Most of these methods were developed for 2D or GUI usability evaluation and have been subsequently extended to support VE evaluation. The usability evaluation methods most commonly found in VE literature include:

- *Cognitive Walkthrough*: an approach to evaluating a user interface based on stepping through common tasks that a user would perform and evaluating the interface's ability to support each step. This approach is intended especially to help understand the usability of a system for first-time or infrequent users, that is, for users in an exploratory learning mode [Polson et al., 1992].
- *Formative Evaluation* (both formal and informal): an observational, empirical evaluation method that assesses user interaction by iteratively placing representative users in task-based scenarios in order to identify usability problems, as well as to assess the design's ability to support user exploration, learning, and task performance [Scriven, 1967; Hix & Hartson, 1993]. Formative evaluations can range from being rather informal, providing mostly qualitative results such as critical incidents, user comments, and general reactions, to being very formal and extensive, producing both qualitative and quantitative (e.g., task timing, errors, etc.) results.
- *Heuristic or Guidelines-Based Expert Evaluation*: a method in which several usability experts separately evaluate a user interface design (probably a prototype) by applying a set of "heuristics" or design guidelines that are relevant. No representative users are involved. Results from the several experts are then combined and ranked to prioritize iterative (re)design of each usability issue discovered [Nielsen & Mack, 1994].
- *Post-hoc Questionnaire*: a written set of questions used to obtain demographic information and views and interests of users after they have participated in a (typically formative) usability evaluation session. Questionnaires are good for collecting subjective data and are often more convenient and more consistent than personal interviews [Hix & Hartson, 1993].

- *Interview / Demo*: a technique for gathering information about users by talking directly to them. An interview can gather more information than a questionnaire and may go into a deeper level of detail. Interviews are good for getting subjective reactions, opinions, and insights into how people reason about issues. “Structured interviews” have a pre-defined set of questions and responses. “Open-ended interviews” permit the respondent (interviewee) to provide additional information, ask broad questions without a fixed set of answers, and explore paths of questioning which may occur to the interviewer spontaneously during the interview [Hix & Hartson, 1993]. Demonstrations (typically of a prototype) may be used in conjunction with user interviews to aid a user in talking about the interface.
- *Summative or Comparative Evaluation* (both formal and informal): an evaluation and statistical comparison of two or more configurations of user interface designs, user interface components, and/or user ITs [Scriven, 1967; Hix & Hartson, 1993]. As with formative evaluation, representative users perform task scenarios as evaluators collect both qualitative and quantitative data. As with formative evaluations, summative evaluations can be formally or informally applied.

There have been several innovative approaches to evaluating VEs that employ one or more of the evaluation methods given above. Some of these approaches are shown in Table 1. This particular set of research literature was chosen to illustrate the wide range of methods and combination of methods available for use.

Research Example	Usability Evaluation Method(s) Employed
[Bowman & Hodges, 1997]	Informal Summative
[Bowman, Johnson, & Hodges, 1999]	Formal Summative, Interview
[Darken & Sibert, 1996]	Summative Evaluation, Post-hoc Questionnaire
[Gabbard, Hix & Swan, 1999] [Hix et. al, 1999]	User Task Analysis, Heuristic Evaluation, Formative Evaluation, Summative Evaluation
[Steed & Tromp, 1998]	Heuristic Evaluation, Cognitive Walkthrough
[Slater, Usoh & Steed, 1995]	Post-hoc Questionnaire

Table 1. Examples of VE usability evaluation from the literature

A closer look at these, and other research efforts, shows that the type of evaluation method(s) used, as well as the manner in which it was extended or applied, varies from study to study. It is not clear whether an evaluation method or set of methods can be reliably and systematically prescribed given the wide range of design goals and user interfaces inherent in VEs. However, it is possible to classify those methods that have been successfully applied to VE evaluation to reveal common and distinctive characteristics among methods.

3.1 Classification of VE usability evaluation methods

We have created a novel classification space for VE usability evaluation methods. The classification space (figure 1) provides a structured means for comparing evaluation methods according to three key characteristics: *involvement of representative users*, *context of evaluation*, and *types of results produced*.

The first characteristic discriminates between those methods that *require* the participation of representative users (to provide design or use-based experiences and options), and those methods that do not (methods not requiring users still require a usability expert). The second characteristic describes the type of context in which the evaluation takes place. In particular, this characteristic identifies those methods that are applied in a generic context and those that are applied in an application-specific context. The context of evaluation inherently imposes restrictions on the applicability and generality of results. Thus, conclusions or results of evaluations conducted in a generic context can typically be applied more broadly (i.e., to more types of interfaces) than results of an application-specific evaluation method, which may be best-suited for applications that are similar in nature. The third characteristic identifies whether or not a given usability evaluation method produces (primarily) qualitative or quantitative results.

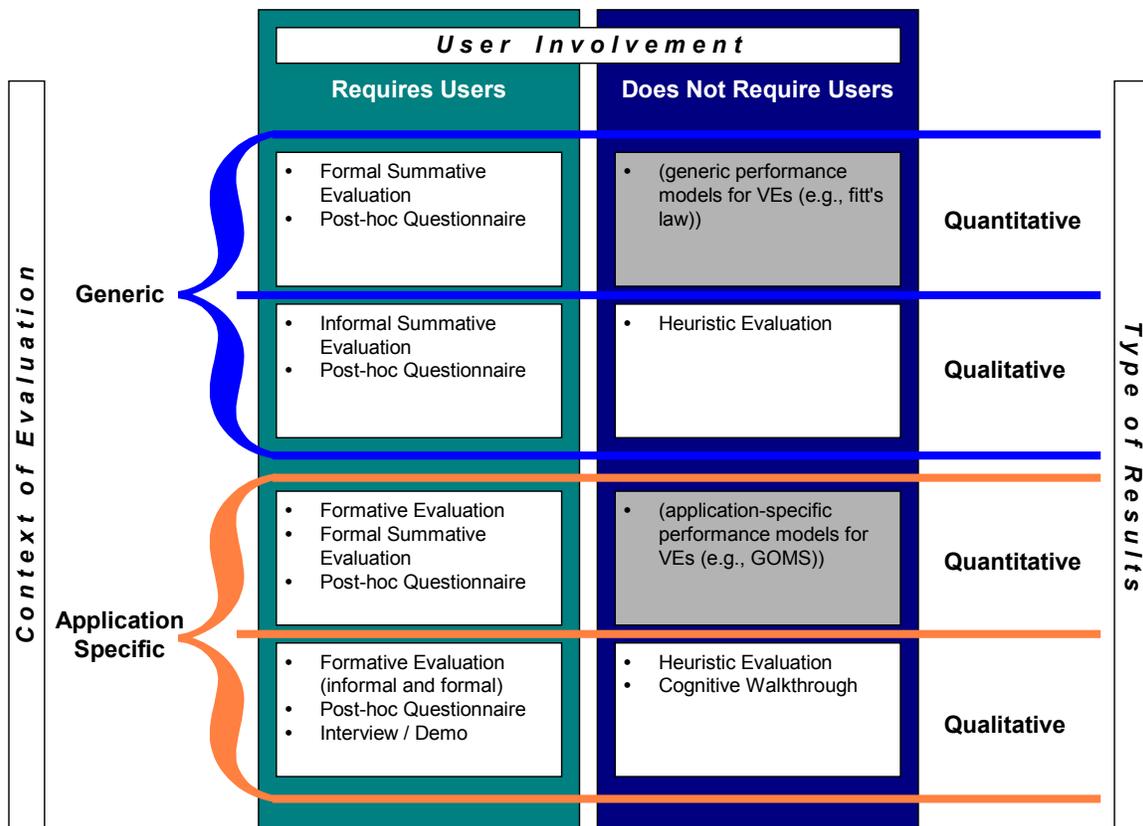


Figure 1. A Classification of Usability Evaluation Methods for VEs

Note that the characteristics described above are not designed to be mutually exclusive, and are instead designed to convey one (of many) usability evaluation method characteristics. For example, a particular usability evaluation method may produce both quantitative and qualitative results. Indeed, many of the identified methods are flexible enough to provide insight at many levels.

Figure 1 shows that there are two areas in which evaluation methods have not been successfully developed or applied (the shaded boxes in figure 1). More specifically, there appear to be no current VE usability evaluation methods that do not require users and that can be applied in a generic context to produce quantitative results (upper right of figure 1). Note that some possible existing 2D and GUI evaluation methods are listed in parentheses, but these have not yet been applied to VEs. Similarly, there appears to be no method that provides quantitative results in an application-specific setting that does not require users (third box down on the right of figure 1). These areas are ripe for further research.

While this classification provides some information about a method's utility, it does not provide enough information to guide application of one or more methods. More specifically, the space does not convey "when" in the software development lifecycle a method is best applied, or "how" several methods may be applied either in parallel or serial. In most cases, the answers to these questions cannot be answered without a comprehensive understanding of each of the methods presented, as well as the specific goals and circumstances of the research or development effort. In the following sections, we present two well-developed VE evaluation approaches, compare them in terms of practical usage and results, and discuss ways they can be integrated for greater power and efficiency.

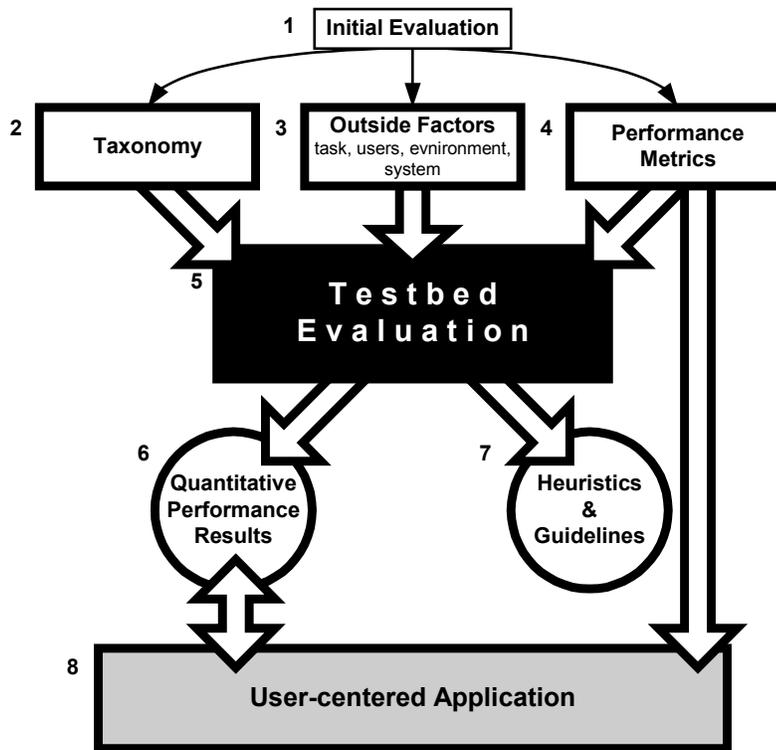


Figure 2. Bowman & Hodges' [1999] Evaluation Approach

3.2 Testbed evaluation approach

Bowman and Hodges [1999] take the approach of empirically evaluating ITs outside the context of applications (i.e., within a generic context, rather than within a specific application), and add the support of a framework for design and evaluation, which we summarize here. Principled, systematic design and evaluation frameworks give formalism and structure to research on interaction, rather than relying solely on experience and intuition. Formal frameworks provide us not only with a greater understanding of the advantages and disadvantages of current techniques, but also with better opportunities to create robust and well-performing new techniques, based on knowledge gained through evaluation. Therefore, this approach follows several important evaluation concepts, elucidated in the following sections. Figure 2 presents an overview of this approach.

The first step towards formalizing the design, evaluation, and application of ITs is to gain an intuitive understanding of the generic interaction tasks in which one is interested, and current techniques available for the tasks (see figure 2, area labeled 1). This is accomplished through experience using ITs and through observation and evaluation of groups of users. These initial evaluation experiences are heavily drawn upon for the processes of building a taxonomy, listing outside influences on performance, and listing performance measures. It is helpful, therefore, to gain as much experience of this type as possible so that good decisions can be made in the next phases of formalization.

The next step is to establish a *taxonomy* (figure 2, 2) of ITs for the interaction task being evaluated. These taxonomies partition a task into separable subtasks, each of which represents a decision that must be made by the designer of a technique. In this sense, a taxonomy is the product of a careful task analysis. Once the task has been decomposed to a sufficiently fine-grained level, the taxonomy is completed by listing possible technique components for accomplishing each of the lowest-level subtasks. An IT is made up of one technique component from each of the lowest-level subtasks. For example, the task of changing an object's color might be made up of three subtasks: selecting an object, choosing a color, and applying the color. The subtask for choosing a color might have two possible technique components: changing the

values of R, G, and B sliders, or touching a point within a 3D color space. The subtasks and their related technique components make up a taxonomy for the object coloring task.

Ideally, the taxonomies established by this approach need to be correct, complete, and general. Any IT that can be conceived for the task should fit within the taxonomy. Thus, subtasks will necessarily be abstract. The taxonomy will also list several possible technique components for each of the subtasks, but they do not list every conceivable component.

Building taxonomies is a good way to understand the low-level makeup of ITs, and to formalize differences between them, but once they are in place, they can also be used in the design process. One can think of a taxonomy not only as a characterization, but also as a design space. Since a taxonomy breaks the task down into separable subtasks, a wide range of designs can be considered quickly, simply by trying different combinations of technique components for each of the subtasks. There is no guarantee that a given combination will make sense as a complete IT, but the systematic nature of the taxonomy makes it easy to generate designs and to reject inappropriate combinations.

ITs cannot be evaluated in a vacuum. A user's performance on an interaction task may depend on a variety of factors (figure 2, 3), of which the IT is but one. In order for the evaluation framework to be complete, such factors must be included explicitly, and used as secondary independent variables in evaluations. Bowman and Hodges identified four categories of outside factors.

First, *task characteristics* are those attributes of the task that may affect user performance, including distance to be traveled or size of the object being manipulated. Second, the approach considers *environment characteristics*, such as the number of obstacles and the level of activity or motion in the VE. *User characteristics*, including cognitive measures such as spatial ability or physical attributes such as arm length, may also contribute to user performance. Finally, *system characteristics* may be significant, such as the lighting model used or the mean frame rate.

This approach is designed to obtain information about human performance in common VE interaction tasks – but what is performance? Speed and accuracy are easy to measure, are quantitative, and are clearly important in the evaluation of ITs, but there are also many other performance metrics (figure 2, 4) to be considered. Thus, this approach also considers more subjective performance values, such as perceived ease of use, ease of learning, and user comfort. For VEs in particular, presence [Witmer & Singer, 1998] might be a valuable measure. The choice of IT could conceivably affect all of these, and they should not be discounted. Also, more than any other current computing paradigm, VEs involve the user's senses and body in the task. Thus, a focus on user-centric performance measures is essential. If an IT does not make good use of human skills, or if it causes fatigue or discomfort, it will not provide overall usability despite its performance in other areas.

Bowman and Hodges use *testbed evaluation* (figure 2, 5) as the final stage in the evaluation of ITs for VE interaction tasks. This approach allows generic, generalizable, and reusable evaluation through the creation of testbeds – environments and tasks that involve all important aspects of a task, that evaluate each component of a technique, that consider outside influences (factors other than the IT) on performance, and that have multiple performance measures. A testbed experiment uses a formal, factorial, experimental design, and normally requires a large number of subjects. If many ITs or outside factors are included in the evaluation, the number of trials per subject can become overly large, so ITs are usually a between-subjects variable (each subject uses only a single IT), while other factors are within-subjects variables. Testbed evaluations have been performed for the tasks of travel and selection/manipulation [Bowman, Johnson, and Hodges, 1999].

Testbed evaluation produces a set of results or models (figure 2, 6) that characterize the usability of an IT for the specified task. Usability is given in terms of multiple performance metrics, with respect to various levels of outside factors. These results become part of a performance database for the interaction task, with more information being added to the database each time a new technique is run through the testbed. These results can also be generalized into heuristics or guidelines (figure 2, 7) that can easily be evaluated and applied by VE developers.

The last step is to apply the performance results to VE applications (figure 2, 8), with the goal of making them more useful and usable. In order to choose ITs for applications appropriately, one must understand the interaction requirements of the application. There is no single “best” technique, because the technique that is best for one application will not be optimal for another application with different requirements. Therefore, applications need to specify their interaction requirements before the most appropriate ITs can be chosen. This specification is done in terms of the performance metrics that have already been defined as part of the formal framework. Once the requirements are in place, the performance results from testbed evaluation can be used to recommend ITs that meet those requirements.

3.3 Sequential evaluation approach

Gabbard, Hix & Swan [1999] present a sequential approach to usability evaluation for specific VE applications. The sequential evaluation approach is a usability engineering approach, and addresses both design and evaluation of VE user interfaces. However, for the scope of this paper, we focus on different types of evaluation and address analysis, design, and prototyping only when they have a direct effect on evaluation.

While some of its components are well-suited for evaluation of generic ITs, the complete sequential evaluation approach employs application-specific guidelines, domain-specific representative users, and application-specific user tasks to produce a usable and useful interface for a particular application. In many cases, results or lessons learned may be applied to other, similar applications (for example, VE applications with similar display or input devices, or with similar types of tasks) and, in other cases (albeit less often), it is possible to abstract the results to generic cases.

Sequential evaluation evolved from iteratively adapting and enhancing existing 2D and GUI usability evaluation methods. In particular, we modified and extended specific methods to account for complex ITs, non-standard and dynamic user interface components, and multimodal tasks inherent in VEs. Moreover, we applied the adapted/extended methods to both streamline the usability engineering process as well as provide sufficient coverage of the usability space. While the name implies that the various methods are applied in sequence, there is considerable opportunity to iterate both within a particular method as well as among methods. It is important to note that all the pieces of this approach have been used for years in GUI usability evaluations. The unique contribution of the Gabbard, Hix & Swan [1999] work is the breadth and depth offered by progressive use of these techniques, adapted when necessary for VE evaluation, in an application-specific context. Further, the way in which each step in the progression informs the next step is an important finding, as discussed near the end of this section.

Figure 3 presents the sequential evaluation approach. It allows developers to improve a VE’s user interface by a combination of expert-based and user-based techniques. This approach is based on sequentially performing user task analysis (see figure 3, area labeled 1), heuristic (or guidelines-based expert) evaluation (figure 3, 2), formative user-centered evaluation (figure 3, 3), and summative comparative

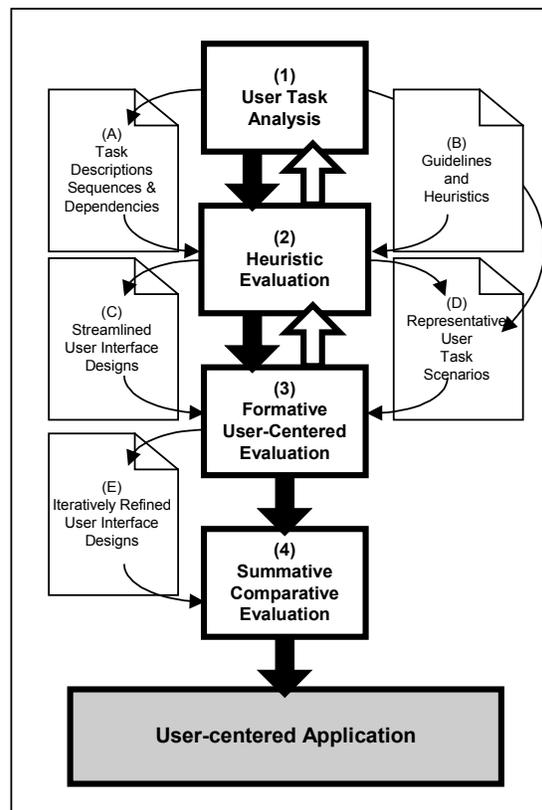


Figure 3. Gabbard, Hix & Swan’s [1999] Sequential Evaluation Approach

evaluations (figure 3, 4), with iteration as appropriate within and among each type of evaluation. This approach leverages the results of each individual method by systematically defining and refining the VE user interface in a cost-effective progression.

Depending upon the nature of the application, this sequential evaluation approach may be applied in a strictly serial approach (as figure 3's solid black arrows illustrate) or iteratively applied (either as a whole or per individual method as figure 3's white arrows illustrate) many times. For example, when used to evaluate a complex command and control battlefield visualization application [Hix et al., 1999], a single user task analysis was followed by significant iterative use of heuristic and formative evaluation, and lastly followed by a single, broad summative evaluation.

From experience, this sequential evaluation approach provides cost-effective assessment and refinement of usability *for a specific VE application*. Obviously, the exact cost and benefit of a particular evaluation effort depends largely on the application's complexity and maturity. In some cases, cost can be managed by performing quick and lightweight formative evaluations (which involve users and thus are typically the most time-consuming to plan and perform). Moreover, by using a "hallway methodology" [Nielsen, 1999], user-based methods can be performed quickly and cost-effectively by simply finding volunteers from within one's own establishment or facility. This approach should only be used as a last resort, or in cases where the representative user class includes just about anyone. When used, care should be taken to ensure that "hallway" users provide a close representative match to the application's ultimate end-users.

Each of the individual methods in the sequential evaluation approach is described in more detail below, with particular attention to how they were adapted for VE evaluations.

3.3.1 User Task Analysis

A *user task analysis* provides the basis for design in terms of what users need to be able to do with the VE application. This analysis generates (among other resources) a list of detailed task descriptions, sequences, and relationships, user work, and information flow (figure 3, A). Typically a user task analysis is provided by a VE's design and development team, based on extensive inputs from representative users. Whenever possible, it is useful for an evaluator to participate in the user task analysis.

The user task analysis also shapes representative user task scenarios (figure 3, D) by defining, ordering, and ranking user tasks and task flow. The accuracy and completeness of a user task analysis directly affects the quality of the subsequent formative and summative evaluations, since these methods typically do not reveal usability problems associated with a specific interaction within the application unless it is included in the user task scenario (and is therefore performed by the user during an evaluation session). Similarly, in order to evaluate how well an application's interface supports high-level information gathering and processing, representative user task scenarios must include more than simply atomic, mechanical- or physical-level tasking, but also high-level cognitive, problem-solving tasking specific to the application domain. This is especially important in VEs, where user tasks generally are inherently more complex, difficult, and unusual than in, for example, many GUIs.

3.3.2 Heuristic Evaluation

A *heuristic evaluation* or *guidelines-based expert evaluation* may be the first assessment of an interaction design based on the user task analysis and application of guidelines for VE user interface design. One of the goals of heuristic evaluation is to simply identify usability problems in the design. Another important goal is to identify the usability problems *early in the development lifecycle* so that they may be addressed, and the redesign iteratively refined and evaluated [Nielsen & Mack, 1994]. In a heuristic evaluation, VE usability experts compare elements of the user interaction design to guidelines or heuristics (figure 3, B), looking for specific situations in which guidelines have been violated, and therefore are potential usability problems. The evaluation is performed by one or (preferably) more usability experts and does not require users. A set of usability guidelines or heuristics that are either general enough to apply to any VE or are tailored for a specific VE is also required.

Heuristic evaluation is extremely useful as it has the potential to identify many major and minor usability problems. Nielsen [1992] found that approximately 80 percent (between 74 percent and 87 percent) of a system's usability problems may be identified when three to five expert evaluators are used. Moreover, the probability of finding a given *major* usability problem may be as great as 71 percent when only three evaluators are used. From experience, heuristic evaluation of VE user interfaces provides similar results; however, the current lack of well-formed guidelines and heuristics for VE user interface design and evaluation reduce the effectiveness of this approach somewhat.

Nonetheless, it is still a very cost-effective method for early assessment of VEs and helps uncover usability problems that, if not discovered via a heuristic evaluation, will very likely be discovered in the much more costly formative evaluation process. In fact, one of the strengths of the sequential evaluation approach is that usability problems identified during heuristic evaluations can be detected and corrected prior to performing formative evaluations. This approach creates a set of streamlined user interface designs (figure 3, C) that may be more rigorously studied in subsequent evaluations. Therefore, this approach leads to formative evaluation that is more cost-effective and efficient than a formative evaluation that is not based on a documented user task analysis or heuristic evaluation. In most cases, this approach avoids the situation where an iteration of formative evaluation is expended simply to expose obvious and glaring usability problems. A formative evaluation following a heuristic evaluation can focus not on the major usability issues, but rather on those more subtle and more difficult-to-recognize issues. This is especially important because of the cost of VE development.

Once both major and minor usability problems are identified, further assessment is needed to understand *how* particular interface components may affect user performance. To focus subsequent evaluations on these identified usability issues, evaluators use results of both the heuristic evaluation and the task analysis as the basis for representative user task scenarios (figure 3, D). For example, if heuristic evaluation identifies a possible mismatch between implementation of a voice recognition system and manipulation of user viewpoint, then scenarios requiring users to manipulate the viewpoint would be included in subsequent formative evaluations.

3.3.3 Formative Evaluation

Formative evaluation [Scriven, 1967] is a type of evaluation that is applied during evolving or formative stages of design to ensure that the design meets its stated objectives and goals. Williges [1984] and Hix & Hartson [1993] extended formative evaluation to support evaluation of GUI user interfaces. The method relies heavily on usage context (e.g., user task, user motivation, etc.) as well as a solid understanding of human-computer interaction (and in the case of VEs, human-VE interaction). The purpose of formative evaluation is to iteratively assess and improve the usability of an evolving user interface design.

A typical formative evaluation cycle may begin with development of user task scenarios that are specifically designed to explore many facets of a user interface design. Task scenarios should provide ample coverage of tasks identified during a user task analysis. Representative users are recruited to work through the task scenarios as evaluators observe and collect data. Experienced usability evaluators follow a structured and scientific approach to data collection, resulting in large volumes of both qualitative and quantitative data. Both types of collected data are equally important parts of the formative evaluation process; quantitative data indicate that a user performance issue is present, qualitative data indicate where (and sometimes why) it occurred.

Collected data are analyzed to identify user interface components that both support and detract from user task performance and user satisfaction. Alternating between formative evaluation and (re)design efforts ultimately leads to an iteratively refined user interface design (figure 3, E). Refining the user interface design such that it efficiently and effectively supports all user tasks leads to a successful, subsequent summative evaluation, because it ensures that each comparison is fair (i.e., each design is as good as it can possibly be in terms of usability).

3.3.4 Summative Evaluation

Summative or comparative evaluation is an assessment and statistical comparison of two or more configurations of user interface designs, user interface components, and/or ITs. Summative evaluation is generally performed after user interface designs (or components) are complete, and is a typical factorial experimental design with multiple independent variables. Summative evaluation enables evaluators to measure and subsequently compare the productivity and cost benefits associated with different user interface designs. Comparing VE user interfaces requires a consistent set of user task scenarios (borrowed and/or refined from the formative evaluation effort), resulting in primarily quantitative data results that compare (on a task by task basis) a design's support for specific user task performance.

A major impact of the formative to summative progression is that results from formative evaluations inform design of summative studies by helping determine appropriate usability characteristics to evaluate and compare in summative studies. There are invariably numerous alternatives that can be considered as factors in a summative evaluation. Formative evaluations typically point out the most important usability characteristics and issues (e.g., those that recur most frequently, those that have the largest impact on user performance and/or satisfaction, etc.). These issues then become strong candidates for inclusion in a summative evaluation. For example, if formative evaluation showed that users have a problem with format or placement of textual information in a heavily graphical display, a summative evaluation could explore alternative ways of presenting such textual information. As another example, if users (or developers) want a number of different display modes, such as stereoscopic and monoscopic, head-tracked and static, landscape view and overhead view of a map, these various configurations can also be the basis of rich comparative studies related to usability.

4 Comparison of approaches

The two major evaluation methods we have presented for VEs – testbed evaluation and sequential evaluation – take quite different approaches to the same problem, namely, how to improve usability in VE applications. At a high level, these approaches can be characterized in the space defined in section 3. Sequential evaluation is done in the context of a particular application and can have both quantitative and qualitative results. Testbed evaluation is done in a generic evaluation context, and usually seeks quantitative results. Both approaches employ users in evaluation.

In this section, we take a more detailed look at the similarities of and differences between these two approaches. We organize this comparison by answering several key questions about each of the methods. Many of these questions can be asked of other evaluation methods, and perhaps *should* be asked prior to designing a usability evaluation. Indeed, answers to these questions may help one identify appropriate evaluation methods given the research, design, or development goals. Future work (by us and others) should attempt to find valid answers to these and other related questions regarding different usability evaluation methods. However, our immediate goal is to understand the general properties, strengths, and weaknesses of each approach so that the two approaches can be integrated into a broader approach, which we present in section 5.

4.1 What are the goals of the approach?

As mentioned above, both approaches have the ultimate goal of improving usability in VE applications. However, there are more specific goals that exhibit differences between the two approaches.

Testbed evaluation has the specific goal of finding generic performance characteristics for VE ITs. This means that we want to understand IT performance in a high-level, abstract way, not in the context of a particular VE application. This goal is important because if achieved, it can lead to wide applicability of the results. In order to do generic evaluation, the testbed approach is limited to general techniques for common, universal tasks. To say this in another way, testbed evaluation is not designed to evaluate special-purpose techniques for specific tasks, such as applying a texture. Rather, it abstracts away from these specifics, using generic properties of the task, user, environment, and system.

Sequential evaluation's immediate goal is to iterate towards a better user interface for a particular application, in this case, a VE application. It looks very closely at particular user tasks of an application to determine which scenarios and ITs should be incorporated. In general, this approach tends to be quite specific, to produce the best possible UI design for a particular application under development.

4.2 When should the approach be used?

By its non-application-specific nature, the testbed approach actually falls completely outside the design cycle of a particular application. Ideally, testbed evaluation should be completed before an application is even a glimmer in the eye of a developer. Since it produces general performance/usability results for ITs, these results can be used as a starting point for the design of new VE applications.

On the other hand, sequential evaluation should be used early and continually throughout the design cycle of a VE application. User task analysis is necessary before the first interface prototypes are built. Heuristics and formative evaluation of a prototype produce recommendations that can be applied to the next iteration of design. Summative evaluations of different design possibilities can be done when the choice of design (e.g., for ITs) is not clear.

The distinct time periods in which testbed evaluation and sequential evaluation are employed suggests that combining the two approaches is possible, and even desirable. Testbed evaluation can first produce a set of general results and guidelines that can serve as an advanced and well-informed starting point for a VE application's user interface design. Sequential evaluation can then refine that initial design in a more application-specific fashion. We expand on this idea in section 5.

4.3 In what situations is the approach useful?

Testbed evaluation allows the researcher to understand detailed performance characteristics of common ITs, especially user performance. It provides a wide range of performance data that may be applicable to a variety of situations. In a development effort that requires a suite of applications with common ITs and interface elements, testbed evaluation could provide a quantitative basis for choosing them, because developers could choose ITs that performed well across the range of tasks, environments, and users in the applications.

As we have said, the sequential evaluation approach should be used throughout the design cycle of a VE application, but it is especially useful in the early stages of development. Because sequential evaluation produces results even on very low-fidelity prototypes or design specifications, a VE application's user interface can be refined much earlier, resulting in greater cost savings. Also, the earlier this approach is used in development, the more time remains for producing design iterations, which results in a better product. This approach also makes the most sense when a user task analysis has been performed. This analysis will suggest task scenarios that make evaluation more meaningful and effective.

4.4 What are the costs of using the approach?

The testbed evaluation approach can be seen as very costly, and is definitely not appropriate for every situation. In certain scenarios, however, its benefits (see section 4.5) can make the extra effort worthwhile. Some of the most important costs associated with testbed evaluation include: difficult experimental design (many independent and dependent variables, where some of the combinations of variables are not testable), experiments requiring large numbers of trials to ensure significant results, and large amounts of time spent running experiments because of the number of subjects and trials. Once an experiment has been conducted, the results may not be as detailed as some developers would like. Since testbed evaluation looks at generic VE situations, information on specific interface details such as labeling, the shape of icons, and so on will not usually be available.

In general, the sequential evaluation approach is less costly than testbed evaluation because it can focus on a particular VE application rather than paying the cost of abstraction. However, some important costs are still associated with this method. Multiple evaluators may be needed. The development of useful task scenarios may take a large amount of effort. Conducting the evaluations themselves may be costly in terms

of time, depending on the complexity of task scenarios. Most importantly, since this is part of an iterative design effort, time spent by developers to incorporate suggested design changes after each round of evaluation must be considered.

4.5 What are the benefits of using the approach?

Since testbed evaluation is so costly, its benefits must be significant before it becomes a useful evaluation method. One such benefit is generality of the results. Since testbed experiments are conducted in a generalized context, the results may be applied many times in many different types of applications. Of course, there is a cost associated with each use of the results, since the developer must decide which results are relevant to a specific VE. Secondly, testbeds for a particular task may be used multiple times. When a new IT is proposed, that technique can be run through the testbed and compared with techniques already evaluated. The same set of subjects is not necessary since testbed evaluation usually uses a between-subjects design. Finally, the generality of the experiments lends itself to development of general guidelines and heuristics. It is more difficult to generalize from experience with a single application.

For a particular application, the sequential evaluation approach can be very beneficial. Although it does not produce reusable results or general principles in the same broad sense as testbed evaluation, it is likely to produce a more refined and usable VE than if the results of testbed evaluation were applied alone. Another of the major benefits of this method relates to its involvement of users in the development process. Since members of the user group take part in many of the evaluations, the VE is more likely to be tailored to their needs, and will result in higher user productivity, reduced user errors, increased user satisfaction, and so on, in actual use. There may be some reuse of results, because other applications may have similar tasks or requirements, or may be able to use refined ITs produced by the process.

4.6 How are the approach's evaluation results applied?

The results of testbed evaluation are applicable to any VE that uses the tasks studied with a testbed. Currently, testbed results are available for some of the most common tasks in VEs: travel and selection/manipulation [Bowman, Johnson, & Hodges, 1999]. The results can be applied in two ways. The first, informal, technique is to use the guidelines produced by testbed evaluation in choosing ITs for an application. A more formal technique uses the requirements of the application (specified in terms of the testbed's performance metrics) to choose the IT closest to those requirements. Both of these approaches should produce a set of ITs for the application that makes it more usable than the same application designed using intuition alone. However, since the results are so general, the VE will almost certainly require further refinement.

Application of results of the sequential evaluation approach is much more straightforward. Heuristic and formative evaluations produce specific suggestions for changes to the user interface or ITs. The result of summative evaluation is an interface or set of ITs that performs the best or is the most usable in a comparative study. In any case, results of the evaluation are tied directly to changes in the interface of the VE application.

5 Integrated evaluation approach

Based on this analysis of the testbed evaluation and sequential evaluation approaches to VE evaluation, we have found that there are many ways in which these approaches can influence and affect one another when used together as part of a broader approach. To this end, we have identified a number of ways that the results of one approach can be used to strengthen and refine the other. These are summarized in figure 4.

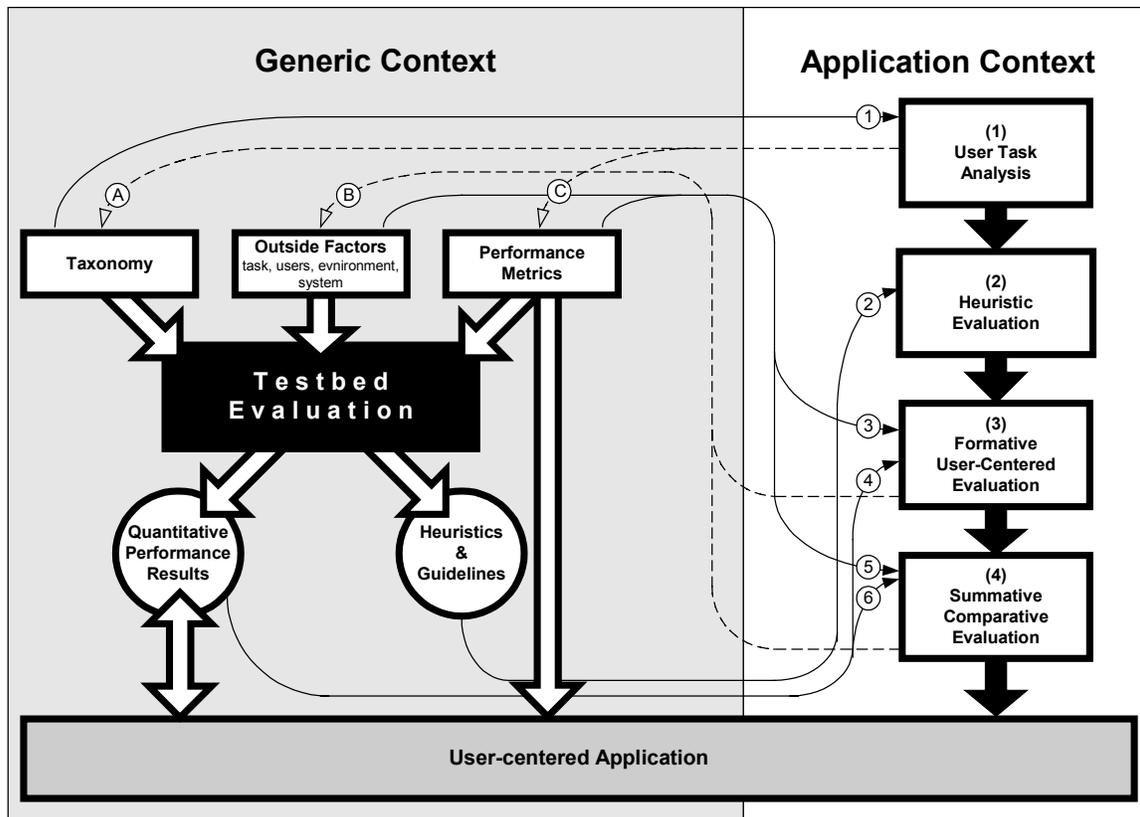


Figure 4. Integration of testbed evaluation and sequential evaluation approaches

As we have noted, there is an inherent separation between the two approaches. Although both have the eventual goal of improving the usability of VE applications, testbed evaluation does this indirectly, through evaluation in a generic context, while the sequential evaluation approach assesses applications directly. However, this does not mean that the two processes are mutually exclusive, or that they are incompatible. On the contrary, we have found many ways the two approaches can influence and benefit one another, and even situations in which they can be used together.

5.1 Testbed evaluation as input to sequential evaluation

There are several ways in which testbed evaluation can affect the sequential evaluation approach. User task analysis, a critical part of the sequential evaluation approach, requires an understanding of tasks users must perform and possible ITs that could be used to accomplish those tasks. Taxonomic structures from the testbed approach provide both of these (see figure 4, area labeled 1). Taxonomies provide a standard way to organize and decompose a task, and they contain a design space from which many ITs can be built.

The general guidelines produced by testbed evaluation can serve as input for heuristic evaluation in the sequential evaluation approach (figure 4, 2). In fact, this addresses a potential problem with using heuristic evaluation for VEs: a lack of heuristics. Since guidelines from the testbed approach are based on experimental evidence, heuristic evaluation using these guidelines should produce a more usable initial design to be fed to the formative evaluation process.

The set of factors other than ITs that could influence performance (outside factors) are an important component of the testbed evaluation process, since they are candidates for independent variables in testbed experiments. For example, one could test whether the number of obstacles in an environment affects the speed of traversing a path in that environment. These same factors can play a role in shaping formative and summative evaluation components of the sequential evaluation approach (figure 4, 3 and 5). The evaluator

can use these factors to more carefully plan task scenarios that assess the range of potential interactions a user could have with the VE. In a similar way, sets of performance metrics defined for testbed evaluation are useful in formative and summative evaluation. These metrics can be checked to ensure that the evaluator observes all variables that contribute to a usable interface.

Finally, quantitative performance results obtained from testbed experiments can play a role in the sequential evaluation process. In formative evaluation (figure 4, 4), an evaluator is trying to produce one or more usable ITs that can later be compared. If testbed results are available for the task in question, incorporation of these ITs into a VE can begin at a much more refined level based on performance results. In the same way, testbed results can help narrow the set of ITs in summative evaluation (figure 4, 6). The relative performance of two ITs may already be known through testbed evaluation, or a particular IT may be known to perform badly in the situation presented by a particular VE application. In any case, these results should be considered before beginning either type of evaluation.

5.2 Sequential evaluation as input to testbed evaluation

Integration of these two approaches can also proceed in the opposite direction, with the sequential evaluation approach serving to inform and refine testbed evaluation. We suggest three ways this might take place. In all three of these cases, the experiences of analyzing a real-world application help to refine the generic model used for testbed evaluation.

One way this can occur involves the process of user task analysis (figure 4, A). Task analysis takes place in the context of a particular application, and can also be refined as the sequential evaluation approach is iterated. This can result in a quite detailed understanding of user tasks, intentions, and mental models for a specific VE. This understanding is exactly what is needed to create good taxonomies of ITs for a particular task, since taxonomies in the testbed approach are based on task decomposition. If taxonomies more closely fit the user's model of a particular task, when this taxonomy is used as a framework for evaluation the results obtained should be a better predictor of user performance in real systems.

Subsequent to the process of user task analysis, usability goals and associated metrics can be determined. It is important for a user to complete tasks efficiently, correctly, without frustration, and in comfort. These characteristics match some of the possible performance metrics given by the testbed approach. However, it is possible that in the process of user task analysis and subsequent setting of usability goals, evaluators will find that a VE has a requirement whose fulfillment cannot be determined using any of the listed performance measures (figure 4, C). The requirement may suggest a new metric to be added to the list and included in future testbed experiments.

It is difficult in the testbed approach to come up with complete lists of the outside factors that could affect performance. This is often done based on intuition alone. However, experiences of evaluators performing formative and summative evaluations can add to and refine these lists (figure 4, B). Evaluators may notice that a user performing a particular task is greatly affected by some characteristic of the environment. This would suggest that this characteristic should be studied in a future testbed experiment to determine the extent of its effects more generally. If that variable has already been studied in a general experiment, it may be possible to give more weight to this factor in analysis of the results.

5.3 Usage scenarios

While the integrated approach described above appears to provide rich coverage of the usability space, we recognize that it is likely too complex and time-consuming to be practically applied to a single VE development effort. Nonetheless, there are research and development arrangements that are well-suited for the integrated approach, including development of a suite of VE applications as well as distributed, asynchronous research and development.

It is reasonable to assume that as VE hardware and user interfaces become more accessible to the mainstream public, there will be significant interest in developing "productivity tools", or software applications that allow users to perform real work, for extended periods, within a VE. Thus, it can be

expected that suites of software applications may be developed that resemble, for example, the Microsoft™ Office suite of tools. In this case, early research and development of common user interface components and user ITs could be furthered by those usability evaluation methods that evaluate in a generic context (such as the testbed evaluation approach). During later stages of research and development, specific applications within the suite could be evaluated using the application-specific evaluation methods (such as the sequential evaluation approach).

But perhaps the most likely scenario in which the integrated method may be applied is in a distributed, asynchronous research setting. In this case, researchers performing generic evaluation of ITs, input/output devices, and user interface components can provide insight, recommendations, and guidelines to the community at large. Subsequently, those performing evaluation of specific applications may use results published from the generic evaluation efforts to aid in their specific application evaluation effort. As described in sections 5.1 and 5.2, the fact that each type of evaluation effort may aid the other introduces the possibility for powerful collaboration among researchers interested in usability evaluation of VEs.

6 Conclusions and future work

Clearly, performing usability evaluation on non-traditional interactive systems requires new approaches, techniques, and insights. While evaluation at its highest level retains the same goals and conceptual foundation, the practical matter of performing actual evaluations can be quite different. We have shown that this is especially true for VEs, and have outlined some of the distinctive characteristics of VE evaluation as well as several possible approaches. This information alone is practical to VE developers and researchers in producing usable applications.

Our integrated methodology for VE evaluation is a novel construct which combines approaches that produce quantitative results with those that produce qualitative results, those that evaluate in a generic context with those that evaluate specific applications, and those that require users with those that do not. By considering all these approaches, evaluators can converge more quickly on a usable system. As we have detailed, each approach brings with it certain advantages that are synergistic when multiple approaches are used.

We plan to continue this work on several fronts. First, we will continue to evaluate real-world VE systems for usability, using the combined approach we describe here. This should lead to a greater understanding of the practical process that can be used to perform evaluation more efficiently and with better results. Second, there are certain VE interaction tasks that have not been explored sufficiently. For example, the task of VE system control, in which the user wishes to issue a command or change the state of the system in some way, is not well-understood. Generic evaluations of various system control techniques would be highly useful to the VE community. Third, we hope others will join us in analyzing usability evaluation methods in terms of the questions posed in section 4. Answers to these and similar questions, for a broader variety of evaluation approaches, can greatly increase the effectiveness and efficiency of performing such evaluations. Such results could help expand the breadth and depth of usability evaluations performed on VE user interfaces. Finally, it is a reality that many VE developers do not choose to perform full usability studies on their systems, making the availability of useful and practical guidelines for VE interface design invaluable. We plan to use our extensive experience in usability evaluation of VEs to create and integrate sets of such guidelines that can be disseminated widely among developers.

Acknowledgments

Portions of this research were funded by the Office of Naval Research, Dr. Helen M. Gigley, Program Manager. Dr. Gigley has funded an on-going collaboration between Virginia Tech and the Naval Research Laboratory in Washington, D.C. for several years. Dr. Ed Swan, of the Naval Research Laboratory, has been a close collaborator on much of this work. Dr. Paul Quinn, also of the Office of Naval Research, has recently provided funding for our efforts. Dr. Richard E. Nance, of Virginia Tech's Systems Research Center, has given much moral support to our research. Dr. Larry F. Hodges of Georgia Tech was instrumental in research on testbed evaluation. We would also like to thank Donald Johnson and Don

Allison of Georgia Tech, and Drew Kessler of Lehigh University for their help and support. We are grateful to all these contributors, without whom this large body of work would not have been possible.

References

- Bowman, D. (2001). Principles for the Design of Performance-Oriented Interaction Techniques. To appear in Stanney, K. (Ed.). *Handbook of Virtual Environment Technology*, Lawrence Erlbaum Associates.
- Bowman, D. and Hodges, L. (1999). Formalizing the Design, Evaluation, and Application of Interaction Techniques for Immersive Virtual Environments. *The Journal of Visual Languages and Computing*, 10(1), 37-53.
- Bowman, D., Johnson, D., and Hodges, L. (1999). Testbed Evaluation of VE Interaction Techniques. Proceedings of the ACM Symposium on Virtual Reality Software and Technology, 26-33.
- Darken, R. P. and Sibert, J. L. (1996). Wayfinding Strategies and Behaviors in Large Virtual Worlds. In *Proceedings of CHI '96: ACM conference on Human Factors in Computing Systems*, 142-149.
- Gabbard J. L. (1997). A Taxonomy of Usability Characteristics for Virtual Environments. Masters Thesis. Department of Computer Science, Virginia Tech.
- Gabbard, J. L., Hix, D, and Swan, E. J. (1999). User Centered Design and Evaluation of Virtual Environments , *IEEE Computer Graphics and Applications*, 19(6), 51-59.
- Hix, D., Swan, E. J., Gabbard, J. L., McGee, M., Durbin, J., and King, T. (1999). User-Centered Design and Evaluation of a Real-Time Battlefield Visualization Virtual Environment. In *Proceedings of IEEE Virtual Reality '99*, 96-103.
- Hix, D. and Hartson, H. R. (1993). *Developing User Interfaces: Ensuring Usability through Product & Process*. New York, John Wiley and Sons.
- Kaur, K. (1998). Designing virtual environments for usability. PhD thesis. Centre for HCI Design, City University, London.
- Kennedy, R.S., Lane, N.E., Berbaum, K.S., and Lilienthal, M.G. (1993). Simulator sickness questionnaire (SSQ): A new method for quantifying simulator sickness. *International Journal of Aviation Psychology*, 3, 203-220.
- Nielsen, J. (1999). Users First: Cheap Usability Tests. Available at: <http://www.zdnet.com/devhead/stories/articles/0,4413,2224316,00.html>.
- Nielsen, J. and Mack, R. L. (1994). Executive summary. In Nielsen, J. & Mack, R. L. (Ed.), *Usability Inspection Methods*, New York, John Wiley & Sons, 1-23.
- Polson, P., Lewis, C., Rieman, J., and Wharton, C. (1992). Cognitive Walkthroughs: A Method for Theory-Based Evaluation of User Interfaces. *International Journal of Man-Machine Studies*, 36, 741-773.
- Poupyrev, I., Weghorst, S., Billinghamurst, M. and Ichikawa, T. (1997). A Framework and Testbed for Studying Manipulation Techniques for Immersive VR. In *Proceedings of VRST '97: ACM Symposium on Virtual Reality Software and Technology*, 21-28.
- Scriven, M. (1967). The methodology of evaluation. In R. E. Stake (Ed.), *Perspectives of curriculum evaluation*, American Educational Research Association Monograph. Chicago, Rand McNally.

Siochi, A. C. and Hix, D. (1991). A Study of Computer-Supported User Interface Evaluation Using Maximal Repeating Pattern Analysis. In *Proceedings of CHI'91 Conference on Human Factors in Computing Systems*.

Slater, M., Usoh, M., and Steed, A. (1995). Taking Steps: The Influence of a Walking Metaphor on Presence in Virtual Reality. *ACM Transactions on Computer Human Interaction*, 2(3) 201-219.

Stanney, K. (1999). Personal communication.

Steed, A. and Tromp, J. (1998). Experiences with the Evaluation of CVE Applications. In *Proceedings of Collaborative Virtual Environments*.

Williges, R. C. (1984). Evaluating Human-Computer Software Interfaces. In *Proceedings of International Conference on Occupational Ergonomics*.

Witmer, B. G. and Singer, M. J. (1998). Measuring Presence in Virtual Environments: A Presence Questionnaire. *PRESENCE: Teleoperators and Virtual Environments*, 7(3), 225-240.

Designing Interactive Theme Park Rides

Lessons Learned Creating Disney's *Pirates of the Caribbean – Battle for the Buccaneer Gold*

Jesse Schell and Joe Shochet
Walt Disney Imagineering VR Studio
1401 Flower Street
Glendale, California 91221
Jesse.Schell@disney.com, Joe.Shochet@disney.com



Abstract

Interactive theme park rides are an unusual breed of entertainment experience. Half video game, half dark ride, interactive rides have their own unique rules about what makes a good show. Disney's *Pirates of the Caribbean – Battle for the Buccaneer Gold* now at DisneyQuest has been called “the best use of VR in an entertainment application – ever”. This paper will discuss the tools, techniques, technology, psychology, and serendipity that made Pirates a hit. It will also outline general guidelines for creating interactive theme park attractions.

Key Ideas

- Interactive theme park rides are not video games, not rides, but a new medium.
- Intuitive user interfaces are crucial for interactive theme park rides.
- Put more emphasis on the real experiences, and less emphasis on virtual ones.
- People go to theme parks in small groups to have shared experiences together. Interactive theme park rides should be designed around this fact.
- Iterative design is crucial when creating new types of interactive experience.

Interactive theme park rides are not video games, not rides, but a new medium.

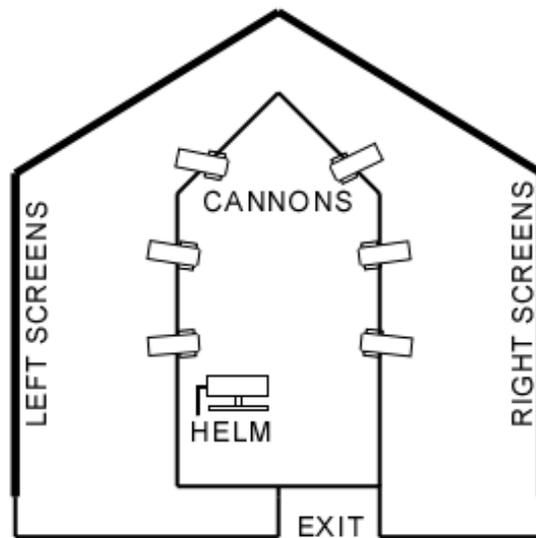
What makes designing interactive theme park rides difficult is that design skills necessary for traditional video games or theme park rides do not always apply in this new medium, and at times can actually work against you. Common video game metaphors such as cut scenes, life meters, restarting levels, and joystick interfaces are not familiar to the average group of Disney theme park goers. The traditional arcade way of learning a game by wasting a few quarters learning the rules and interface and then pumping more quarters in to continue is not feasible with an hour line waiting to play.

Because these rides are interactive, the guest is in control of their own destiny. This means the game needs to reward success and punish failure. Typical theme park attractions do not have this design constraint. On a Disney attraction, even losing must be entertaining.

The fact that *Pirates* is interactive and virtual carries with it an expectation that it will be a video game. Many guests, especially parents, have an anxiety towards video games, believing that only kids will understand or do well at them. *Pirates* overcomes this expectation with a fun, novel interface and game system that does not require any previous gaming knowledge.

Pirates Overview

Pirates is an interactive theme park ride based on the classic *Pirates of the Caribbean* attraction at Disneyland. With themes and inspiration taken from the ride, this virtual interactive experience treats four guests to an overwhelming immersive adventure on the high seas. With one guest steering at a real helm, the other three guests man six real cannons to defeat virtual enemy pirate ships, forts, sea monsters and ghostly skeletons to collect and defend as much gold as possible in the five minute experience. *Pirates* uses wrap-around 3D screens, 3D surround sound, and a motion platform boat to fully engage the guest as a pirate. Currently *Pirates* is open at DisneyQuest, Disney's virtual theme park venue, in Orlando and Chicago.



Layout of ship and screens

Interactive rides: A delicate balance

At every turn, the design of *Pirates* was driven by the need to balance between letting the guests have control over their adventure, and making sure that each adventure is a great one. Here are the solutions we found to some of the problems created by this balancing act.

Problem: The captain might steer the ship to dull places.

We solved this problem with several techniques:

- **“Architectural Weenies”**
“Weenie” is phrase coined by Walt Disney himself. It refers to the technique used on movie sets of guiding stage dogs by holding up part of a sausage. The classic “weenie” is the castle at Disneyland. It draws the eye, and the eye draws the feet, and people walk to the castle at the center of the park. In the case of *Pirates*, we had three main “weenies”, one for each island: a volcano, an enormous fort, and a plume of smoke coming from a burning town. No matter which way the boat is facing, at least one of these “weenies” is in view. Since the coolest action takes place at the islands, we want to guide the captains to go there.
- **Guide Ships**
Since the short-term goal of the game is to fire on other pirate ships, captains strive to get near these ships so that their gunners can get a clear shot. Many of the ships in the *Pirates* world are “on their way” to the islands mentioned above. Many captains, in just trying to stay near these ships find that just as they have destroyed the ship, they have arrived at one of the islands, without even trying to get there.
- **Sneak attacks**
But what if the captain ignores the guide ships? Even if he heads toward one of the “weenies” it might mean as long as a minute during which the gunners have little to shoot at. For this reason, we created special “sneak attack” ships that “magically” appear behind the players ship, and quickly pull up along side, when no other boats are in range.
- **“The Waterspout”**
This was our nickname for our “last ditch” forcefield that surrounds the gameplay area. If a captain tries to sail out of the main gameplay area and out to open sea, they hit the forcefield, and the ship is “magically” pointed back to where the action is. The few guests who see this don’t even realize that anything unusual has happened. They are just pleased to have their boat going somewhere cool.

Problem: The pacing of the adventure needs to grow in excitement and build to a climax, while still making the guests feel in control of their destiny.

The initial hook of the adventure is in the form of a non-interactive sequence where Jolly Roger the Ghost Pirate explains the roles of the captain and gunners, encourages the players to sink

many pirate ships in order to get their gold, and then does a 3D close up gag, followed by a motion base gag. After that, the guests are in complete control, and the pacing of the show is mostly governed by the weenies, the guide ships, and the sneak attacks. These combine to give a nice balance between action, and short periods of calm. Guests fight the guide ships, the sneak attack ships, and the ships in other interesting encounters at the islands. Each island is a scenario, with a little story and a couple secrets:

In the “burning town” island, guests fight other pirate ships while sailing through a narrow canal with buildings and frantic townspeople on either side. At the end of the canal an enemy ship loaded with dynamite blocks the way.

In the “volcano” island, guests fight other pirate ships, but can also get bonus treasure by firing on the “treasure troves” on shore. This scenario has two possible endings. Either the volcano blows up (and blows you back out to sea) or the captain discovers the secret waterfall lagoon, which ends with the ship going over a waterfall, but “magically” falling back to the main gameplay area.

In the “fort” island, guests are attacked with fireballs by soldiers at the fort. An enormous gold ship (hard to sink, but worth many points) is just setting sail, guarded by navy ships.

There is only time to visit one or two of these islands in the five minute adventure, which lends to replay value.

To make the journey from one island to another more exciting than just a long sequence of sneak attack ships, we introduced a sea serpent, who attacks the ship. We timed his appearance carefully, so that some variety is provided just when it is needed. Most guests mistakenly believe that they “found him”, which is great, because it is exactly what we want them to think.

We couldn’t figure out how to guarantee the guests would find their way to an exciting climax, so we made the climax come to them. Jolly Roger (the host from the beginning) appears suddenly after four and a half minutes, and it turns out that he only encouraged you to do battle and gather gold so that he could steal it from you. A battle against Jolly Roger’s ghost ship and dozens of flying skeletons then ensues as our ship races past jagged rocks. The host turning out to be the villain is a great surprise for the guests, and provides great storytelling economy, as one character wears two important hats. The experience ends one of two ways: either the guests defeat Jolly Roger, and enter a victory lagoon where they can now shoot fireworks from their cannons, or Jolly Roger defeats the guests, and our boat explodes as giant skulls swirl around us, and we sink to the bottom of the ocean where sharks swim over our wreckage.

Both endings are exciting, but the lose ending is really the more exciting one, to help compensate for the fact that the guests just lost the game. This way, even if you lose, you feel pretty good, because the whole thing was just so cool.

Intuitive user interfaces are crucial for interactive theme park rides

In order to ensure the high throughput that theme parks demand, there must be no time wasted acclimating the guest to the story, interface, or game rules. One thing *Pirates* makes extensive use of is an incredibly rich back-story that every guest can relate to – that of being a pirate. The attraction title, music, and theming of the queue line immediately gets the guest in the correct mind-set to play. They know what to expect, what is expected of them, and can then focus on the details of the interface and game rules.

The physical interface must be easy to learn and easy to use the first time a guest plays. *Pirates* uses very simple, obvious interfaces like a steering wheel to steer and actual cannons to point and shoot virtual cannonballs. We decided to make the helm and cannons active while the guests are boarding the ship. This gives them a few seconds to fire off a test shot or try a turn on the wheel to acclimate to the interface before the pressure of the actual game begins. Extensive guest testing of the interface assured us the design would work with real guests.

Aside from physical interface, the communication between the guest and the game elements must be intuitive. We chose to bend reality in places where it would make the game easier to adapt to and play. Some examples include:

- We exaggerated the virtual cannonball color to an unexpected light blue color because it contrasted with most other colors in the game and thus made the cannonballs easier to see. We changed the cannonball physics during the final scene of the game to be attached to the ship because the ship moves, bumps and turns too much to keep track of your cannonball otherwise.
- In the opening scene Jolly Roger delivers an introduction on the left side of the ship. We found many guests looking to the right would not realize he was even onscreen so we slowly darkened the rightmost screens to encourage the guests to look in the direction of Jolly Roger.
- The captain's throttle can move the boat at about 90 miles per hour and turn on a dime because actual boat physics would have resulted in a very slow and boring game.
- Instead of programming what the optimal strategy for an enemy pirate ship to defeat the guest would be, the enemies were developed with rules that would provide a good show. Some examples include:
 - Staying broadside with the guest ship
 - Attacking evenly on both sides of the guest ship
 - Keeping pace with the guest ship
 - Leading guests from the relatively low action open seas into high action scripted scenarios at the islands
 - Sneaking up from offscreen when the guests had nothing to shoot at
 - Staying away from the guest ship while the serpent was onstage

By choosing to be less concerned with reality and more concerned with what was fun, we created an experience that matches guests' expectations of what being a Pirate might feel like. Therefore it is easier to adapt to, quicker to learn, and is a better show.

More emphasis on the real experiences, less emphasis on virtual.

To be successful, the ride must extend beyond what guests can get elsewhere. With the power of graphics supercomputers in video game consoles in the home, these rides simply cannot keep up with the curve to remain fresh from a visual standpoint. To be worth the price of admission, these rides must overwhelm, play to more senses, and provide a real physical experience that cannot be replicated in the living room. In *Pirates*, the use of a motion base gives guests a unique experience of feeling every cannonball hit, every wave, and the bites of attacking sea monsters. Localized 3D surround sound and tactile speakers create a wide sound bed of cannonballs whizzing by, crew yelling from the rigging, and boat creaks underfoot. Strobe lights help create the explosion of a direct cannonball hit on the helm. 3D stereo glasses not only put the action in your face, but also make the projector screens disappear, creating a very convincing virtual world.

Because guests must run from cannon to cannon to best defend their ship they get a physical experience instead of merely sitting passively in front of a monitor. Guests get social interaction from bumping into each other, taking turns on cannons, barking out orders, and negotiating the rocking ship. The feeling of being tired and practically out of breath after five minutes of plundering with your friends or family is a feeling that you got your money's worth.

In the cannon interface we had a problem that guests could fire the cannons too fast, sometimes more than 5 shots per second thus trashing the enemies before the other players could even get a chance. Software timers to keep the number of shots down created frustration because the cannon was not responding to the guest input. Instead we created haptic blocks to keep the number of shots low. By introducing some weight and friction into the firing mechanism (a pull string) it is physically hard to shoot more than once or twice per second. By solving the problem with real physical methods instead of arbitrary virtual software blocks, the game remains fair and playable. For the ambitious player with enough energy to still shoot a ridiculous number of shots per second, each rapid fire shot decreases in power after the first few shots. This keeps the game balanced between the casual players and the hard core shooters.

No one goes to a theme park alone.

In order to create a successful interactive theme park ride, it is essential to understand the mindset of the guests who will be experiencing it. While many people play video games as a way to have a rewarding solo experience free from social pressures, people almost never go to theme parks alone. Instead, they go in small groups, with the intention of enjoying shared experiences together. Many Location Based Entertainment titles suffer from failure to understand this fact. *Pirates* turns out to be a powerful shared experience, and one that many kinds of small groups can enjoy in different ways.

- Boys enjoy it in the obvious way, as an “adventure and battle fantasy” where they can pilot a pirate ship, and man powerful cannons. While they enjoy some communication, they stay very focused on the task of defeating the bad guys as skillfully as possible.
- Girls also enjoy it, but in a different way. The girls tend to help each other more, and talk back and forth between themselves a bit more. They seem to really enjoy the notion of “banding together” to protect themselves against a common enemy.

- Mothers enjoy it in ways that pleasantly surprised us. Generally, mothers at amusement parks are less interested in having a good time themselves, and are more interested in making sure the rest of the family has a good time. Piloting the pirate ship becomes an ideal task for them, because it not only affords them a good view of the rest of the family enjoying the experience, but also allows them to tune the experience (by steering the boat appropriately) to maximize fun for the family.

Two other factors strengthened the social interaction of *Pirates*:

- Shared visual and audio displays. Unlike some VR group experiences where each player has their own monitor and speakers, in *Pirates* all the players can be sure that all the other players are hearing the same things and are looking at things from nearly the same point of view. As a result, players communicate with each other in natural ways (shouting, gesturing, and pointing), that might prove less useful with independent displays.
- Shared input devices (cannons). We designed the game so that the gunners needed to move around the ship to effectively deal with the enemies. The sheer physicality of running around the ship, bumping into other players, and quickly negotiating who uses what gun when, provides a fun and natural social interaction amongst the group.

One potentially “cool” feature was cut early because of the negative effects it was having on social interaction. The original design called for a separate “map” monitor that only the captain could see. The idea was that by looking at the interactive map (which would show the entire area, and track the enemies as they moved) the captain would be able to better know where to next steer the ship, and not have to rely on nearby visual cues. The hope was that the captain would be able to use the map as a “God’s eye view”, glancing at it occasionally to get a better sense of what was all around him, planning where to go next, and warning the gunners of sudden attacks from behind.

Early prototypes of the interactive map showed that captains didn’t use it this way at all. Instead, they either ignored it completely, or (more often) became immersed in it, playing the whole game while staring at the map, and not looking out to sea at all. This created a significant communication disconnect between the captain and the gunners. Very often a captain “steering by the map” would confuse the gunners by shouting for them to fire on ships that he thought were good targets, but in reality were behind the ship, or out of range.

Sessions would begin with a lot of:

“Shoot him!”

“Shoot who?”

“The ship on the right!”

“What ship on the right? There’s nothing there!”

and end up with the captain giving up on the map, or (even worse) giving up trying to communicate with the gunners.

Getting rid of the map put the gunners and the captain in the same visual space, allowing them to have useful conversations. And while we were initially concerned that the captain “wouldn’t have enough to do” compared to the gunners (which was part of the reason we introduced the map), it quickly became clear that in trying to:

- steer the ship around nearby obstacles
- navigate to distant destinations
- position the ship so that the gunners could get the best shots at enemies
- keep an eye on both sides of the ship to watch out for “surprise attacks”
- alert the gunners about these attacks

the captain had plenty to do, and he seemed to have a lot more fun doing it than when the “captain’s map” was there to distract him.

Iterative design is crucial when creating new types of interactive experience.

As with any new medium, paving the frontier is a game of trial and error. To reduce the risk of designing ourselves too far down dead ends during development, we use an iterative design strategy. For *Pirates* we mocked up the basic show concept – steering a ship and shooting cannons at enemy ships – in less than two months. We were able to do this because we reused existing in-house software tools and hardware systems developed on previous DisneyQuest attractions. It was important to us to mock up the fun first. By using temp sounds and quick artwork prototypes we learned early on what dynamics were important to make the game work. This left almost an entire year to concentrate on perfecting the balance, script, artwork, and audio without worrying too much about the underlying game dynamics.

Throughout the development of the project we used an interpreted scripting environment written in Scheme that allowed us to reprogram the game while the attraction was running live, even while guest testers were in the middle of the game. This allowed for rapid iteration of ship behavior, cannon parameters, difficulty settings, and general development of the game logic.

Guest testing is always an important part of the process when developing interactive entertainment. Guest testing assures that the designer’s assumptions are checked, game systems are properly balanced, and unpredicted social behaviors can be understood and used to enhance the game. By testing early and often, unpredicted issues can be dealt with eliminating expensive redesigns.

Disney's Aladdin: First Steps Toward Storytelling in Virtual Reality

Randy Pausch¹, Jon Snoddy², Robert Taylor², Scott Watson², Eric Haseltine²

¹University of Virginia

²Walt Disney Imagineering

Figure 1: A Guest's View of the Virtual Environment

ABSTRACT

Disney Imagineering has developed a high-fidelity virtual reality (VR) attraction where guests fly a magic carpet through a virtual world based on the animated film "Aladdin." Unlike most existing work on VR, which has focused on hardware and systems software, we assumed high fidelity and focused on using VR as a new medium to tell stories. We fielded our system at EPCOT Center for a period of fourteen months and conducted controlled experiments, observing the reactions of over 45,000 guests.

contact author: Randy Pausch, Computer Science Department, Thornton Hall, University of Virginia, Charlottesville, VA 22903. Pausch@virginia.edu, 804/982-2211

Riders filled out an exit survey after the experience, and with select groups we used a number of other data-gathering techniques, including interviews and mechanically logging where guests looked and flew.

Our major finding is that in a high fidelity VR experience, men and women of all ages suspend disbelief and accept the illusion that they are in a different place. We have found that in VR, as in all media, content matters. Novices are unimpressed with the technology for its own sake; they care about what there is to do in the virtual world. We can improve the experience by telling a pre-immersion "background story" and by giving the guest a concrete goal to perform in the virtual environment. Our eventual goal is to develop the lexicon for this new storytelling medium: the set of communication techniques shared between directors and the audience. We conclude with a discussion of our second version of the Aladdin project, which contains a large number of synthetic characters and a narrative story line.



INTRODUCTION

Most existing work on virtual reality (VR) has focused on hardware and system software [1, 3, 5, 6, 7, 10, 12, 23, 24]. The price of a high quality system has placed it out of reach for most people interested in content. Building high quality, low cost VR systems is important, but we believe the exciting challenge in VR is learning what to do with the medium.

We believe that the content questions are the really hard ones. The goal of this project has been to allow the content producers, or authors, to assume the existence of satisfactory technology and to focus directly on authoring in the new medium of VR. We produced high-quality content based on flying a magic carpet in the animated film “Aladdin” [2]. Figure 1 shows a screen shot from the system.

We field-tested the system on over 45,000 guests at EPCOT Center. In this paper we report our detailed observations, the guests’ exit surveys, and data we recorded during guest experiences. This is not a systems implementation paper; we describe the hardware and software only as context for describing the guest experience. In addition to guest experiences, we also describe industrial design solutions to the problems of high volume usage. In early 1996, we will deploy a second version with a narrative story line and a large number of reactive characters. We conclude with lessons learned from creating virtual environments and characters for our second version, especially controlling the narrative in an interactive medium.

Our underlying premise is that VR is a new medium, as film, radio, and television once were. As motion pictures matured, directors and audiences developed a lexicon including close ups, cross cuts, flash backs, etc. Over time a common language, or lexicon, will evolve for VR; this project is our first step towards that goal.

SYSTEM DESCRIPTION

In each of our field trials, four guests donned head-mounted displays and piloted a flying carpet. Because they were running on separate systems, the pilots could neither see nor interact with each other.

We designed the system for robustness, high volume usage, and high accessibility. Unlike research setups, theme park equipment is used extensively, continuously, and abusively. Failures with a one-in-a-million chance of happening can occur once a week in a typical theme park attraction.

The Head Mounted Display

The system used an internally developed head-mounted display (HMD), shown in Figure 2. The two main design constraints were to provide high image quality and to make it easy to put the HMD on quickly, to support the high throughput of guests in a theme park attraction. In early trials we learned that having adjustments such as a focus knob on the HMD confused guests, since they had no baseline to distinguish between high and low image quality. Therefore, we designed a system that would accommodate a large variation in where a guest’s eyes sit with respect to the optics.

Figure 2: The Head Mounted Display

Image quality considerations drove us to use CRTs instead of LCDs, a tradeoff that increased the HMD’s weight and extended its center of mass. We partially compensated for this by providing spring-suspension of the HMD from the ceiling. Major design challenges in the HMD included avoiding visible pixel boundaries, obtaining high contrast, minimizing inter-ocular rivalry, and addressing the weight balance and packaging issues. For head-tracking, we used a magnetic position/orientation tracker.

Unlike many other VR systems, our HMD display was bi-ocular, not stereo. We rendered a single, horizontally wide graphics window and fed partially overlapping view windows to each of the CRTs in the HMD. For applications such as ours, stereoscopy is surprisingly unimportant as a depth cue [8].

We addressed hygiene issues by having the HMD snap onto a per-guest inner “cap” that can be cleaned separately. The inner liner also allowed us to adjust tightness to each guest’s head before monopolizing the more expensive HMD and image generator. The HMD fit comfortably over eyeglasses; the only notable issue was guests with hair tied in buns.

Sound

The HMD contained two speakers that rested close to, but not in physical contact with, the guest’s ears. We used a combination of stereo ambient sound, binaural recorded sound, and eight channels of localized sound. We recorded the binaural sound track via a high quality binaural head (essentially, microphones placed in a mannequin head). The binaural soundtrack included background voices, animals, and other “clutter” sounds. We recorded multiple binaural tracks, and then mixed those layers to form a composite recording. When the binaural recording was played during the VR experience, even though those sounds “moved with the head,” they established a believable background sound field. It is in this context that the eight special channels were convolved to localize in real-time based on head tracking [26]. The localized channels provided main characters and large sound

effects. The stereo sound (primarily music) established emotional context, the binaural sound established the believable three-dimensional space, and the localized sounds gave strong, specific cues for orientation. The three levels increasingly traded recording quality for localization, and the binaural and localized sounds worked well together because they employed the same head-related transfer functions [4].

Seating, Controls, and Motion Base

Guests were seated straddling a motorcycle-style seat as shown in Figure 3. A benefit of this design is that the guests were firmly grounded, with weight on their buttocks, knees, and feet. Additionally, this design accommodated a wide range of heights. Guests gripped a steering mechanism representing the front of a magic carpet. Turning left or right controlled yaw of the carpet, and tilting controlled the pitch of the carpet. Imagine a car's steering wheel; pulling the top of the wheel toward the driver pitched the carpet up, pushing it pitched the carpet down. Pushing the entire mechanism forward controlled velocity. Figure 4 shows a schematic diagram of the carpet controls.

Figure 3: The Physical Setup

Figure 4: Schematic Diagram of Carpet Controls

We mounted the seat on a movable base that pitched up and down in response to the steering control. Originally, the motion base also tilted side-to-side, but this caused discomfort during early testing so we removed the side-to-side tilt. Surprisingly, the presence or absence of a motion base had no substantial effect on guest satisfaction, or anything else we measured with exit surveys.

An early version of the system simulated wind with a rate-controlled fan blowing air over the guests. Much to our disappointment, most guests wearing the HMD did not notice it.

Image Generation

For each guest, we used a custom Silicon Graphics computer with 512 megabytes of RAM, 16 megabytes of texture memory, eight 150 MHz MIPS R4400 CPUs and three Reality Engine pipelines with four RM5 raster managers each. We rendered 20 frames per second on each pipe, interleaving the frames to achieve a 60 Hz frame rate. Although the frame rates could vary between 15 and 60 during a flight, the overwhelming majority of the time the system rendered at 60 Hz.

Because hardware lighting can draw attention to edges in models with low polygon count, our artists decided to render all polygons with hand-painted textures, with no hardware lighting. This also improves rendering time slightly, but we did it for image quality, not speed.

Model Management And Show Programming

A custom software system, called the *player*, provided scene management and character animation playback. The player provided a Scheme interface on top of a C/C++ layer, all on top of SGI Performer [19].

The player used Performer's support for multiple levels of detail. Unlike a flight simulator, most of our scene was close, so we used only two levels of detail per object. Artists created both models for each object because degrading a model "by hand" still produces better results than automatic means [20]. We sometimes used large texture "flats" for distant objects, and switched to three dimensional models as the guest approached.

Programming of various show elements, such as an object's reaction when hit by the carpet, was performed in the topmost layer of the player, a locally developed "Story Animation Language." This SAL layer implemented cooperative lightweight threads on top of Chez Scheme [9], an incrementally compiled Scheme.

In our second version, the database is much larger, and is partitioned into distinct scenes. The player software pre-fetches geometry and texture maps as guests fly from one scene to another [11]. Between scenes, we include explicit "transition areas," such as hallways and caverns. Transition areas have a smaller number of polygons, which buys us time to pre-fetch textures. Transitions bend and twist, thus ensuring that at no point can the guest quickly look back to the previous scene.

GUEST SELECTION

We deployed the system at EPCOT Center in Orlando, Florida, from July 1, 1994 until September 8, 1995. Every twenty minutes a group of up to 120 guests was given a brief technical lecture about VR followed by a demonstration where four guests were selected to "fly a magic carpet."

The attraction was intentionally hidden in a remote area of the park. Most guests entered not because they had a strong interest in VR, but because our attraction was "the next thing" to do. Guests could not volunteer to fly; they were selected by the ride operators. The operators maintained a strict policy of avoiding guests who showed an active interest in VR. Therefore, rather than pertaining to a small subset of VR enthusiasts, we believe that our results are essentially a fair cross section of the theme park population. Some guests did decline the invitation to fly. Interviews revealed this was primarily due to stage fright, not an aversion to trying VR.

The selected pilots did not hear the technical lecture about VR. We gave them a background story that they would be stepping into the feature film "Aladdin." We instructed them that the main goal was to have fun, but that they were also searching for a character from the film. The marketplace scene was chosen because it 1) contains familiar objects such as doors which establish scale, 2) is a brightly lit daytime scene, and 3) contains wide variety, encouraging exploration. There was typically time for a one-to-three minute practice flight followed by a few minutes of rest before the audience entered and the four minute flight began.

NOVICES' EXPERIENCES

We exposed a large, non-self selected, population of guests to a high-fidelity virtual experience in a controlled environment. At least one other system has exposed large numbers of novices to VR [25]. However, Virtuality's users were self-

selected. Their users wanted to try VR, and paid for the experience. Our sample is much more diverse.

Our findings are drawn from a variety of sources, including written post-flight guest surveys, logged flight data, extensive conversations with the day-to-day attraction operators, observations of guests' flights, and interviews of guests before, during and after their flights.

Technologists should be aware that most guests were not impressed by the technology itself; guests assumed VR was possible and had an expectation of extremely high quality. Many had seen the "holodeck" on Star Trek, and expected that level of quality. Once in a virtual environment, guests focused on what there was to do in the virtual world – content matters!

General Observations

We were able to sustain the illusion that the guests were in another place. Men and women of all ages suspended disbelief and a large number reported the sensation that they were "in" the scene. This is hard to conclude from exit surveys, but guests also provided unsolicited cues, such as panicking or ducking their heads as they approached obstacles.

Guests cared about the experience, not the technology. Most guests had no concept of how VR works, nor did they care. They focused on the sensation, which was exhilarating for most guests. Many guests shouted "Wow!" or "Whee!" in their first thirty seconds.

The experience was overwhelming. Between sensory overload and the task of trying to control the carpet's flight, many guests were so cognitively taxed that they had trouble answering questions early in their flights.

Guests needed a goal. If not given a specific goal, guests would ask "What should I be doing?"

Guests needed a background story. We found that giving as much context as possible about the scene helped reduce the severity of the transition from the real to the virtual environment. *Background story* is the set of expectations, goals, major characters, and set of rules that apply to the virtual world. Ironically, in lower fidelity, less believable VR systems, this need for background story may not be as evident. We believe it is the abrupt transition into a *believable* virtual world that is problematic. Performing a good transition from the real to the virtual world is an open challenge.

Guests liked exploring, and seeing new spaces. Most guests did not spend much time studying detail in a given place; they tended to move on quickly to new vistas.

Guests did not turn their heads very much. This could be because they were piloting a vehicle, or because they were not accustomed to being able to turn their heads when looking at displayed images. For many, we believe the latter. Guests often watched characters "walk out of frame," as would happen with television or movies. Our strongest indication came from many pilots where we waited 90 seconds into their flight, then explicitly told them to turn their heads. At that point, they clearly had the "aha" of the head-tracking

experience. While we suspect that different content would be more conducive to head turning, head tracking is far enough from most guests' experiences with film and television that we suspect this will be a problem for many systems.

Controlling the carpet was a problem for many guests. This prompted the addition of test flights before the show began. Many guests flew out into the desert or up above the city to find a space where there were fewer obstacles, making flight easier. Although we could have had the magic carpet fly itself, our surveys indicated that the control and freedom are important parts of the experience. Six-axis control is a very difficult problem and an important design challenge is finding appropriate control constraints.

VR must be personally experienced. In addition to the 45,000 guests who piloted carpets, we had over one million audience members who observed the pilots' progress on display monitors. The audience members enjoyed the show and understood that *something* fascinating was going on with the pilots, but it was clear that VR is foreign enough that most people can not fully comprehend it without direct personal experience. Audience members often asked if the pilots could see or interact with each other.

Presence and Immersion

Although it is difficult to formally measure, we believe that most guests suspended disbelief and had the experience of being in a new place. Our choice of an animated world underscored that believability is different from photo-realism. In fact, we reject the term "simulation," as we provide an experience not possible in reality. Our virtual environment was not realistic, but it was consistent with the large number of animated worlds that guests had seen before. Guests flew, but had no fear of heights; guests reacted to the characters, but were not afraid of a guard who brandished a sword. In many ways, this environment was compelling without being disturbing.

A common sight in a 3-D theater is to see large numbers of guests reaching out to grab the projected image. We speculate that they are compelled to conduct this test because their perceptual and cognitive systems are in conflict; their eyes tell them the object is within arm's length, but their brain tells them it is just a projection. In our system, we saw no evidence of the need to test. Guests did not intentionally run into objects to see if the objects really existed. In fact, guests did the opposite, often involuntarily ducking when they felt they could not avoid a collision.

In general, we believe that the need for high fidelity can be reduced by engaging the user in a complex, real-time task. For example, the desktop DOOM game [14] and the SIMNET tank simulator [18] both get users to the point where the interface becomes transparent and the user focuses on task performance, which requires a sense of presence. Our system did so with the mildest of tasks, that of searching for a character. At first, we suspected that the difficult task of piloting the carpet might lower our fidelity requirements. Therefore, we ran experiments where the carpet flew itself. During those tests guests achieved the same suspension of disbelief, with the only task being to look around. Our metric for suspension of disbelief was their reactions to the environment, such as ducking when flying near objects.

What produced the effect of immersion is difficult to know. Even for guests who did not turn their heads much, the HMD physically blocked out the real world. Sound was also very important, as many guests remarked that the sound of wind when they flew high, or the crashing noises when they ran into walls strongly reinforced their sense of being there. In post flight interviews, guests told us that their illusion of presence was destroyed when characters did not react.

Reaction to Virtual Characters

It is more difficult to build a believable character than a believable scene. Although our major focus was on building the environments, we were pleased that a few of our guests did respond to characters. The show began with instructions from a parrot who told the pilots to nod their heads. Some guests actually heeded his command. Another character covered his head and shouted "Don't you have a horn on that thing?!" when guests flew near him. Many guests shouted back at this character. One young girl finished the attraction in tears because she had spent several minutes attempting to apologize to him, but instead continually triggered hostile responses whenever she approached him. (All the characters had a small set of dialog sequences that could be triggered).

The key to a successful character is the suspension of disbelief; one must talk to the puppet, not the puppeteer. Most guests flew at high speed, zooming past the characters. When guests did slow down, they expected the characters to respond and were very disappointed when the characters did not. At the very least, characters should orient their heads and eyes and look at the guest. Our next system is incorporating this feature.

We suspect that the limited believability of our first system's characters is due to low fidelity. All characters in the first show, such as those shown in Figure 5, were animated with motion capture, where sensors recorded an actor's body motions in real time, and those values were used to drive the animation. Our second version uses higher quality key frame animation. While testing of the second version is not yet complete, early indications are that we will cross a fidelity threshold in character animation much as this project crossed one in environment fidelity.

Figure 5: Animated Characters

Men vs. Women

One of our original objectives was to discover whether VR appealed only to the narrow (primarily young male) video game market, or was more like feature films, appealing to males and females of all ages. While *content* will still matter, the technology itself did not turn away any guests. On post flight surveys, the reaction of both genders and all age groups was almost identical on all questions. One major difference was that many women are afraid that they would not be able to operate the equipment properly. This surfaced both as a pre-flight concern and as a post-flight comparison. They often asked how they performed relative to the other pilots. Also, during in-flight interviewing men were more likely to talk about the technology, whereas women were more likely to talk about the experience and emotional impact. Neither men nor women complained about having to wear the HMD.

VR for the Disabled

Everyone involved with the project noted the impact on both the pilots and the audience when motion-impaired guests flew. Accessibility is a fundamental design constraint at Disney parks, and we have a substantial wheelchair population. One of our four stations could be converted for wheelchair access in about ten seconds, and we had several wheelchair fliers per day. The sense of mobility and the joy it brought them was overwhelming.

Motion Sickness

We did not find motion sickness to be as significant an issue as we had feared. During selection, we asked guests if they were prone to motion sickness, and warned that they might feel motion sick during the experience. We also told them they could stop at any time and remove the HMD. Post flight surveys indicated that, as with many theme park attractions, some guests reported discomfort or dizziness, but they mostly described it as a mild sensation. We do not know if guests who felt discomfort or dizziness self-limited their head motion; our logged data showed no such correlation. Reports of discomfort went up when the room was warmer, which is consistent with discomfort reports from platform-based simulator rides. We were careful to limit the duration of the experience. As with any "thrill" experience, discomfort increases with ride length.

GUEST POST-RIDE SURVEYS

After their flights, we asked guests to complete a one page survey with about five multiple choice questions. Guests were identified on the survey only by first name, and over 95 percent of the guests completed a survey. Most who declined did so because of low English skills. We asked many questions and report here a relevant subset. Our sample was 48.5 percent female, and included all ages.

We tried to ask questions that would yield different responses by gender and age. However, we were unable to design questions where the responses were not reasonably consistent across all groups. Thus, we conclude that VR experiences have broad appeal. Responses are presented here by gender (M = male, F = female); breakdown by age is equally similar. The possible responses are listed in the same order as they

appeared on the printed survey form. Because we made ongoing changes to the surveys, the number of total responses to any question is variable -- after each question is the total number of responses.

What did you LIKE the most? (N=25,038)

	all	M	F
characters	11%	10%	12%
helmet fit	4%	4%	3%
motion	32%	32%	32%
picture quality	17%	19%	15%
sound	8%	7%	9%
steering control	21%	21%	21%
town	7%	7%	7%

What did you DISLIKE the most? (N=22,479)

	all	M	F
characters	5%	6%	4%
helmet fit	20%	20%	20%
motion	13%	14%	13%
picture quality	13%	13%	13%
sound	6%	6%	6%
steering control	34%	33%	36%
town	8%	8%	7%

Guest rating of the Experience (N=1,903)

	all	M	F
terrible	1%	1%	1%
okay	4%	4%	5%
good	11%	9%	13%
great	54%	49%	57%
best thing at Disney	23%	28%	20%
best thing in my life	7%	9%	5%

As an absolute answer, we take this with a grain of salt. It is unlikely that our system is really the best thing in seven percent of our guests' lives. However, the scale is useful for comparing males and females; again, we found an overwhelming similarity.

Would You Recommend it To a Friend? (N=273)

	all	M	F
yes	99%	100%	98%
no	1%	0%	2%

It Made Me Feel Like I Was... (N=1,336)

	all	M	F
visiting a town	14%	15%	14%
playing a video game	23%	19%	25%
being inside a movie	45%	49%	43%
in the middle of a dream	17%	16%	17%
invisible	1%	1%	2%

Had You Heard About Virtual Reality Before Today? (N=307)

	all	M	F
no	16%	12%	18%
I had read about it	36%	37%	34%
seen on TV or movies	49%	50%	47%

On My Next Ride, I Would Most Like To... (N=324)

	all	M	F
--	-----	---	---

see more characters	35%		32%	40%
see more towns/places	38%		37%	38%
see the other pilots	27%		31%	22%

The Best Thing About it Was... (N=439)

	all		M	F
the characters	5%		3%	6%
flying	42%		41%	43%
exploring/seeing new things	23%		23%	23%
being able to go where I wanted	30%		33%	28%

I would most like to... (N=426)

	all		M	F
have the carpet fly itself	9%		5%	12%
fly the carpet myself	84%		90%	80%
ride while a friend is flying	6%		4%	8%

LOGGED DATA

For over two thousand guests we recorded the position and orientation of the pilot’s head and the carpet twenty times each second. Our original hope was that we could see patterns of where guests flew and what they found interesting. In fact, we discovered that guests flew almost indiscriminately; no obvious patterns of travel emerged from the data.

The analysis of head turning data was more interesting. Our first question was “How much do guests turn their heads?” The data confirmed what many researchers describe as the dirty secret of VR. In many scenarios, people in HMDs do not turn their heads very much. Figure 6 shows a top-view polar histogram of head yaw; for a guest facing right, the length of each line shows the proportional amount of time spent at each angle. Figure 7 shows a conventional histogram of guest head yaw; the height of each bar is the portion of total time spent at that angle.

Figure 6: Polar Histogram of Head Yaw

Figure 7: Conventional Histogram of Head Yaw

Figure 8 shows that the difference between male and female head yaw is negligible. In fact, every category that we examined (gender, age, which lab technician instructed them, whether or not they experienced motion discomfort, how much they enjoyed the ride) yielded essentially the same profile.

Figure 8: Male vs. Female Head Yaw

Figure 9 shows that head pitch, or up/down tilt, is even more confined than head yaw.

Figure 9: Head Pitch

We were not surprised that guests looked straight ahead most of the time. However, we were surprised by the following: instead of portion of total time, Figure 10 graphs the widest yaw angle guests *ever* experienced. One way to read this graph is that 90% of the guests never looked more than 75 degrees to either side. One could infer that building a 150 degree wide, screen-based display would be as good as an HMD for 90% of the guests. That conclusion would ignore that the HMD field of view must be added to the head yaw, and that the HMD also prevents visual intrusion from the real world.

TELLING STORIES IN VR

Given that VR can present a compelling illusion, researchers can and should pursue its uses for education, training, medical applications, games, and many other purposes. As a storytelling company, we are focusing on using VR as a storytelling medium.

Figure 10: Widest Yaw Angles Seen by Various Guest Percentages

Script vs. Guest Controlled Cameras

Our first system was the first of many steps towards telling stories in VR. Our next show contains over twenty scenes, approximately fifteen high-fidelity characters, and a narrative story line that includes the ability to alter the sequence of events. Our guest assumes a role in an immersive feature film. The major challenge of allowing the guest to become a character is that the director gives up control of the narrative. While this is true of many interactive, non-HMD based games, the problem becomes acute with an HMD.

Because we let the guest control the viewpoint we must build characters and scenes that look good from all vantage points. By establishing entrances to scenes we control the initial view of each scene, a technique used in well-designed theme parks. The inability to cut from scene to scene or view to view is very frustrating for content authors. We have experimented with having characters that are attached to the guest's head, and appear to be hanging off the front of the HMD. This allows us to interject a brief "scene" including that character.

There is an intrinsic conflict between a pre-constructed narrative and a guest-controlled exploration. An interactive system can dynamically re-configure the story to avoid

omission of critical portions. As our director said, "It's as if you decide to leave a movie early, and the projectionist edits the film to make sure you see the important ending before you leave."

In other perceptually intensive theme park attractions such as effects-laden stereoscope films or platform simulators, we have learned to keep the story line simple and clear. We must do the same for VR. Our initial experience indicates that VR is good at placing guests in an environment, and we look forward to seeing its storytelling capacities evolve.

All our experiences to date have been with a novice audience. Filmmakers once used devices such as tearing away calendar pages to show flashbacks or passage of time. As the audience became more experienced, these devices became unnecessary.

Controlling the Narrative

We are fairly successful at composing a scene that draws the guest's attention to a desired spot. We have also experimented with using characters to direct attention. In some scenes characters point where we want the guest to look, and in others, we have a character move to be in line with another object we want in view. All these techniques can be quickly tried; the key is to test them on novices.

We have experimented with explicit techniques for controlling the guest's position, such as having a character grab the carpet and drag the guest to a desired location. Another coarse grain technique is to close doors behind guests to keep them from back-tracking. We have also experimented with implicit techniques such as a "water skiing tow rope" metaphor [13], where an invisible boat is controlling the eventual position and the guest is free to fly within a moving envelope.

Sound

In films, the sound track, particularly the musical score, tends to carry the emotional tone for most scenes. Because we no longer control timing we must choose sound tracks that work with wide variation in duration, and we must be able to make the transition smoothly from one ambient sound to the next based on guest actions.

Many VR system architects are concerned with the underlying technology for localizing sound. In our experience, the careful selection/creation of ambient sounds and music, i.e. the content, is much more important than the specific details, or even the use of, sound localization.

AUTHORING

In the process of building our first show we have learned a number of lessons regarding the process of authoring in VR.

Rapid prototyping is essential for authoring. Flight simulator technologies often guarantee rendering frame rates, but require long (many hour) periods to change the show's content. Our SAL/Scheme layer allows code interpretation at run time, similar to MR/OML [22], Alice [16], and World Toolkit [21].

We could not have developed the show without this interpreted layer.

Character animation in our second system approaches the look of traditional animation. This is not surprising, since the principles of animation apply regardless of the medium [15]. The key to achieving this was involving the artists in the development of the underlying technology, rather treating it as a given. We can now generate a new scene or character animation in under a week.

The fidelity trap for VR is that unlike many other media, a low-quality “quick and dirty” mockup is often misleading. Since a partial or low-quality mockup may not yield accurate results when we test guests on it, we often must build systems to completion before we know what works well. This is partially because there are not yet good tools for sketching three dimensional scenes and animations.

Motion capture vs. key frame animation: Our first system used motion capture to animate characters. This allowed us to produce a large amount of animation quickly, but the quality was not as high as the key frame-based animation we are using for the second version. Motion capture is troublesome for non-human characters, often seems too realistic, and requires laborious post-processing of the data.

Branching story: In a linear narrative, a character’s behavior is completely pre-planned. When the guest’s actions can cause a character to perform different pre-animated sequences, we refer to that as a *branch*. In the original show most characters performed a repose animation until the guest approached, and then branched to perform a reaction animation. While this makes an interesting and active scene, in most cases it does not provide enough different branches to allow the guest to easily suspend disbelief.

Autonomous characters vs. authored branching: Artificially intelligent characters are an interesting concept, but it will be a long time before they are believable in any but the simplest background role. For the next few years, we feel that believable character performances will be made up of branches of pre-animated material rather than computer programs for several reasons: 1) “thinking” characters are far enough into the future to be off the planning horizon, 2) characters who can construct decent sounding sentences are not much closer, 3) a good animator can achieve a much more believable dramatic performance than a computer program, and 4) even simple branching works when properly done.

In our second version characters have multiple possible behaviors that are triggered by context. Even our simplest characters have a default behavior, a reaction to the guest’s presence, and a reaction to the guest’s departure. A major technical challenge is to smoothly blend between various pre-defined animations [17].

Rotation of characters to face guests is important to present the illusion that characters are real. We find that first turning a character’s head, then his or her body, works best. The technical challenge is to avoid bad interactions between the automatic rotation and the character’s key frame animation.

RESEARCH CHALLENGES

Based on our experiences, we present the following as open challenges to the research community:

- 1) Finding mechanisms that allow guests to self-calibrate the intensity of the experience. Currently we must keep the experience tame enough to be enjoyable for our more sensitive guests.
- 2) Developing constraints to solve the six-degree-of-freedom problems in controlling flight; i.e. navigation and motion through virtual spaces.
- 3) Development of software to better support animators, especially in the sketching phase. Animators use onion skin paper to superimpose views from multiple frames; this ability is lacking in most software tools.
- 4) The automatic generation of mouth animation from sound source. This is currently labor intensive and not particularly creative work.

CONCLUSIONS

This project gave over 45,000 people a first exposure to virtual reality (VR). While we have made what we consider to be substantial advances in HMD and rendering technology, our major advances have been in learning how to create and present compelling virtual environments. We stress that this is an exercise that requires both artistic and engineering talent and creativity.

Our guests completed written surveys, and with subsets we logged head and carpet motions. Based on that data and interviews before, during and after guest flights, we conclude that:

Guests suspend disbelief. The illusion is compelling enough that most guests accept being in a synthetically generated environment and focus on the environment, not the technology.

VR appeals to everyone. Both genders and all ages had similar responses to our attraction. This leads us to conclude that VR is like feature films in that different content may segment the market, but the basic technology does not. We also note that wheelchair guests find mobility within VR extremely exciting.

VR must be personally experienced. VR is foreign enough that most people can not comprehend it without direct personal experience.

Fidelity matters. To get most guests to suspend disbelief requires extremely high fidelity. We provide 60 frames per second (at the expense of stereo), for polygonal models with hand-painted texture maps, and we do not use hardware lighting. Texture quality matters much more than polygon count.

Content matters. People love the experience of VR, but even at high fidelity VR by itself is not enough. The public, unlike

the developers, is not impressed with the technology. In fact, the public *assumes* that high fidelity VR exists and immediately focuses on what there is to do in the synthetic environment.

The illusion of presence is fragile. Although guests suspend disbelief, inconsistencies can *instantly* shatter the illusion. For example, objects inter-penetrating, or characters not responding to the guest's presence completely shatter the sense of presence.

Guests need a background story. VR is an overwhelming experience of being thrust into a new environment. A good way to soften this transition is to provide a background story that familiarizes the guest with the new environment before the immersion. This is a standard technique in theme park attractions, typically provided in a pre-show.

Guests need a goal. Guests need to know why they are in the virtual world and what they are supposed to do.

Guests do not turn their heads much. We were surprised at how little people turned their heads in this flight-based experience. We attribute this to the mass of the HMD, the need to look where one is flying, and guests' inexperience with a head-tracked medium.

Input controls are hard. We developed a novel input mechanism for controlling flight. Since no one flies magic carpets in the real world we could not transfer everyday skills. After many design iterations we believe that six axis control is a phenomenally difficult problem and conclude that designers must limit degrees of freedom.

Tell a straightforward story. As we have learned with other intensive media, such as effects laden stereoscopic films and motion-base simulators, when the guest is perceptually overwhelmed it helps to keep the story short and clear.

Aladdin is a beginning, not an end. Our original goal was to move past the technology. Our first system produces a compelling illusion and our next efforts are to examine whether we can tell stories in this new medium. Our second version of the project, scheduled for release in early 1996, contains a large number of characters and a narrative story line.

ACKNOWLEDGMENTS

This work is the effort of many talented people over several years; we mention here only a subset, but express our gratitude to all involved.

Special thanks, in alphabetical order, to: Daniele Colajacomo, for managing the character modeling in the EPCOT show; Dave Fink, for helping start the project; Phillip Freer and Gary Daines, for their art direction and world design; Andy Ogden, for his industrial design on the steering and HMD; George Scribner, for his work on story and character in the EPCOT show; and Dave Spencer, for his management of the EPCOT show installation.

We thank all the other artists and engineers who worked on this project, and we would especially like to express our deepest gratitude to the families and significant others who supported these individuals in their efforts.

We would also like to thank Evans & Sutherland, Silicon Graphics, NASA Ames Research Center, the staff who ran the attraction at EPCOT Center, and Matt Conway.

REFERENCES

- [1] Jon Airey, John Rohlf, Frederick Brooks, Towards Image Realism with Interactive Update Rates in Complex Virtual Building Environments, ACM SIGGRAPH Special Issue on 1990 Symposium on Interactive 3D Graphics 24:2, 1990, pages 41-50.
- [2] Walt Disney Home Video. Distributed by Buena Vista Home Video, Dept. CS, Burbank, CA, 91521. ISBN 1-55890-663-0. Originally released in 1992 as a motion picture.
- [3] Chuck Blanchard, Scott Burgess, Young Harvill, Jaron Lanier, Ann Lasko, Reality Built for Two: A Virtual Reality Tool, ACM SIGGRAPH 1990 Symposium on Interactive 3D Graphics, March 1990.
- [4] J. P. Blauert, Spatial Hearing, MIT Press, Cambridge, MA, 1983.
- [5] Frederick Brooks, Walkthrough -- A Dynamic Graphics System for Simulating Virtual Buildings, Proceedings of the 1986 Workshop on 3D Graphics, Chapel Hill, NC, October 23-24, 1986, ACM, pages 9-21.
- [6] Steve Bryson, Creon Levit, The Virtual Wind Tunnel, IEEE Computer Graphics and Applications, July 1992, pages 25-34.
- [7] Christopher Codella et al., Interactive Simulation in a Multi-Person Virtual World, Proceedings of the ACM SIGCHI Human Factors in Computer Systems Conference, May 1992, pages 329-334.
- [8] James Cutting and Peter Vishton, Perceiving Layout and Knowing Distances: The Integration, Relative Potency, and Contextual Use of Different Information About Depth, Handbook of Perception and Cognition: Perception of Space and Motion, Vol. 5, Academic Press (to appear).
- [9] R. Kent Dybvig. The Scheme Programming Language. Prentice-Hall, 1987.
- [10] S. S. Fisher, A. M. McGreevy, J. Humphries, W. Robinett, Virtual Environment Display System, Proceedings on the 1986 Workshop on Interactive 3D Graphics, pages 77-87, October 23-24, 1986.
- [11] Thomas Funkhouser, Carlo Sequin, Seth Teller, Management of Large Amounts of Data in Interactive Building Walkthroughs, Proceedings of the 1992 ACM

Symposium on Interactive Three-Dimensional Graphics, April, 1992, pages 11-20.

7801-7805 Mesquite End Drive, Suite 105, Irving, Texas 75063, USA 001-214-556-1800, 1-800-ILLUSION, enquiries@tx.virtuality.com.

- [12] Thomas Furness, The Super Cockpit and Human Factors Challenges, Human Interface Technology (HIT) Laboratory of the Washington Technology Center, Tech Report HITL-M-886-1, October, 1986.
- [13] Tinsley A. Galyean, Guided Navigation of Virtual Environments, 1995 ACM Symposium on Interactive 3D Graphics, April 1995, pages 103-104, Monterey, CA.
- [14] Id Software, Inc.; information available via <http://www.idsoftware.com/>.
- [15] John Lasseter, Principles of Traditional Animation Applied to 3D Computer Animation, Computer Graphics (SIGGRAPH '87 Proceedings) Volume 21, number 4, pages 35-44, July 1987.
- [16] Randy Pausch, et al, A Brief Architectural Overview of Alice, a Rapid Prototyping System for Virtual Reality, IEEE Computer Graphics and Applications, May 1995.
- [17] Ken Perlin, Real Time Responsive Animation with Personality, IEEE Transactions on Visualization and Computer Graphics, Vol. 1, No. 1.
- [18] A. Pope, BBN Report No 7102. The SIMNET Network and Protocols. BBN Systems and Technologies, Cambridge, Massachusetts, 1989.
- [19] John Rohlf and James Helman, IRIS Performer: A High Performance Multiprocessing Toolkit for Real-Time 3D Graphics, SIGGRAPH '94 Conference Proceedings, Computer Graphics, July, 1994.
- [20] William J. Schroeder, Jonathan A. Zarge, William E. Lorenson, Decimation of Triangle Meshes, Computer Graphics (SIGGRAPH '92 Proceedings) Volume 26, pages 65-70, July 1992.
- [21] Sense8 Corporation, 100 Shoreline Highway, Suite 282, Mill Valley, CA 94941, 415/331-6318, <http://www.sense8.com/>, info@sense8.com.
- [22] Chris Shaw, Mark Green, Jiandong Liang, Yunqi Sun, Decoupled Simulation in Virtual Reality with the MR Toolkit, ACM Transactions on Information Systems, 11:3, pages 287-317, July 1993.
- [23] Ivan Sutherland, The Ultimate Display, Proceedings of IFIP (International Federation of Information Processing) '65, Vol. 2, pages 506-508.
- [24] Ivan Sutherland, A Head-mounted Three-dimensional Display, Proceedings of the Fall Joint Computer Conference, AFIPS, Vol. 33, pages 757-764.
- [25] Virtuality, <http://www.virtuality.com>, UK office: Virtuality House, 3 Oswin Road, Brailsford Industrial Park, Leicester LE3 1HR, United Kingdom, Tel: +44(0) 116 233 7000, enquiries@virtuality.com. USA office: [26] E. Wenzel, F. Wightman, S. Fisher, A Virtual Display System for Conveying Three-dimensional Acoustic Information, Proceedings of the Human Factors Society, 32nd Annual Meeting, 1988, pages 86-90.



Figure 1: A Guest's View of the Virtual Environment



Figure 2: The Head Mounted Display

High-resolution TIFF versions of these images can be found on the CD-ROM in
S96PR/papers/pausch



Figure 3: The Physical Setup



Figure 5: Animated Characters

High-resolution TIFF versions of these images can be found on the CD-ROM in
[S96PR/papers/pausch](#)

Part III: Annotated Bibliography

20th Century 3DUI Bib: Annotated Bibliography of 3D User Interfaces of the 20th Century

Compiled by Ivan Poupyrev and Ernst Kruijff, 1999, 2000, 3rd revision

Contributors: Bowman, D., Billingham, M., Cugini, J., Dachsel, R., Hinckley, K., LaViola, J.,
Lindeman, R., Pierce, J., Steed, A., Stuerzlinger, W.

to submit contributions to the bib e-mail to poup@mic.atr.co.jp or ernst.kruijff@archit.uni-weimar.de

1. Adelstein, B., Johnston, E., Ellis, S., A testbed for Characterizing Dynamic Response of Virtual Environment Spatial Sensors. *Proceedings of UIST'92*. 1992. ACM. pp. 15.
Keywords: *input devices, sensor lag, spatial sensors, system calibration*
Annotations: This paper describes a testbed for measuring the latency of spatial sensors, but unlike Liang et al. [UIST'91 paper] it does not suggest specific filtering methods. Unlike previous related work, this study measures the performance of the sensor alone. Factors such as code execution time, inter-process communication time, and rendering time do not distort the results.
2. Agronin, M., The Design of a Nine-String Six-Degree-of-Freedom Force-Feedback Joystick for Telemanipulation. *Proceedings of NASA Workshop on Space Telerobotics, 1987*. 1987. pp. 341-348.
Keywords: *haptics, force feedback, joystick, manipulation, teleoperation, telerobotics, virtual reality*
Annotations: Haptic Displays: A six-degree of freedom force feedback joystick. paper basically explains the joystick and how it works. Goes through the physics equations for its motion very painlessly.
3. Angus, I., Sowizral, H., Embedding the 2D Interaction Metaphor in a Real 3D Virtual Environment. *Proceedings of Stereoscopic Displays and Virtual Reality Systems, 2409*. 1995. SPIE. pp. 282-293.
Keywords: *pen & tablet metaphor, constrained interaction, 3D interface, virtual reality, 2D metaphor*
Annotations: The first published system that presented a standard 2D GUI within a 3D virtual environment. Used to allow Web browsing in a VE.
4. Angus, I., Sowizral, H., VRMosaic: web access from within a virtual environment. *IEEE Computer Graphics & Applications*, 1996. 16(3): pp. 6-10.
Keywords: *pen & tablet metaphor, constrained interaction, 3D interface, virtual reality, 2D metaphor*
Annotations: A journal version of the system reported in 1995 at the SPIE conference
5. Ayers, M., Zeleznik, R., The Lego Interface Toolkit. *Proceedings of UIST'96*. 1996. ACM. pp. 97-98.
Keywords: *lego blocks, rapid input device prototyping, 3D input device, 3D interfaces, virtual reality*
Annotations: A novel approach to rapid prototyping of interaction devices for 3D interaction and virtual environments. The devices are built out of the Lego (tm) building blocks with sensors mounted on them simply by snapping them together.
6. Badler, N., Manoochehri, K., Baraff, D., Multi-Dimensional Input Techniques and Articulated Figure Positioning by Multiple Constraints. *Proceedings of Workshop on Interactive 3D Graphics*. 1986. ACM. pp. 151-169.
Keywords: *multi-dimensional input, jacks, constraints, clutching, physical props*
Annotations: This paper describes an attempt to add multi-dimensional input, using a Polhemus tracker, to an early version of Badler's "Jack" articulated figure positioning system. The Polhemus ("wand") was used in two modes: absolute and relative. Absolute positioning was fatiguing. Relative motion allowed the user to move the wand (by releasing the button) when an uncomfortable position was reached. Orientation was always absolute. The implementers thought that the consistent coordinate systems of the wand and their "test scene" would allow intuitive movement, but this was not true. Lack of depth perception ("spatial feedback") on the 2D display made it difficult to select a target; also, simultaneously positioning and orienting the wand proved to be challenging. They tried decoupling wand parameters, but results were still not satisfactory. Using the wand to position a virtual camera was more successful but it was still a consciously calculated process. The implementers found that using a real object as a spatial reference for 3D wand interactions yielded a "natural and effortless" interface. The real object provides the true depth information lacking in the 2D display.
7. Balakrishnan, R., Fitzmaurice, G., Kurtenbach, G., Singh, K., Exploring Interactive Curve and Surface Manipulation Using a Bend and Twist Sensitive Input Strip. *Proceedings of Symposium on Interactive 3D Graphics*. 1999. ACM. pp. 111-118.
Keywords: *input devices, bimanual input, ShapeTape, gestures, curves, surfaces*
Annotations: This paper describes input device made of a continuous bend and twist sensitive strip that is used to edit and create 3D curves and surfaces. In addition the paper also talks about other interaction techniques including command access and system control.
8. Beaten, R., DeHoff, R., An Evaluation of Input Devices for 3-D Computer Display Workstations. *Proceedings of The International Society for Optical Engineering 1987*. 1987. SPIE. pp. 237-244.
Keywords: *input devices, user study, stereoscopic cues, manipulation, positioning, trackball*
Annotations: Describes a user study (16 subjects) testing a 3D positioning task using 3D trackball (free space movements), mouse (three buttons used as mode control for motion in the three orthogonal planes), and a custom thumbwheel device (three wheels, one-handed control,

- arranged to correspond to orientation of display's coordinate system). Output strategies were: perspective encoding of depth and field-sequential stereoscopic encoding of depth. Thumbwheels yielded a more than two-fold increase in positioning accuracy as compared to the other devices. The stereoscopic display reduced positioning error by about 60%. Also, the relative differences between input devices varied across the display conditions, but in general positioning accuracy increased 51-60% with the stereoscopic display. Positioning time: The time associated with the mouse was longer than the other two devices. Positioning with either the trackball or the thumbwheels was about 23% faster.
9. Begault, D., *3D Sound For Virtual Reality and Multimedia*. 1994: Academic Press. pp. .
Keywords: *virtual auditory space, 3D sound, spatial hearing, HRTF, VR, multimedia*
Annotations: This book is one of the first on 3D sound for virtual reality. It provides introductions to the psychoacoustics of spatial sound and implementing virtual acoustics through digital signal processing.
 10. Bier, E., Skitters and Jacks: Interactive 3D Positioning Tools. *Proceedings of Workshop on Interactive 3D Graphics*. 1986. ACM. pp. 183-196.
Keywords: *placement techniques, cursor, skitter, jack, 3D object placement, desktop 3D interfaces*
Annotations: Describes an early version of Bier's "Gargoyle 3D" system. The interactive techniques are primarily geared towards scene composition, including precise placement of objects using affine transforms. Anchors: A "hot spot" used, for example, to select an axis of rotation. End conditions: e. g., the number of degrees to rotate. Jacks: Cartesian coordinate frames used to describe anchors & end conditions. Skitter: 3D cursor (interactively positioned Jack). Uses a gravity function for effective 3D point selection.
 11. Bier, E., Snap-dragging in three dimensions. *Proceedings of Symposium on Interactive 3D Graphics*. 1990. ACM. pp. 193-204.
Keywords: *3D interfaces, constraints, manipulation, desktop 3D, mouse and keyboard, gravity, scene composition*
Annotations: The paper describes an interface introducing "snapping" of interface elements to each other in the context of 3D interactive scene creation. Gravity functions are used to implement snapping, for example a 3D cursor can snap to various elements of the scene., making it easier to select scene elements. System also uses various alignment objects, e.g. lines, planes and spheres which other scene objects can snap to and which can be used as guides for manipulating object.
 12. Billinghurst, M., Put That Where? Voice and Gesture at the Graphic Interface. *Computer Graphics*, 1998. 32(4): pp. 60-63.
Keywords: *multimodal interaction, 3D interfaces, manipulation, voice input, gesture input*
Annotations: Survey of various issues in multimodal interaction with 3D user interfaces in mind.
 13. Billinghurst, M., Baldis, S., Matheson, L., Phillips, M., 3D palette, a virtual reality content creation tool. *Proceedings of VRST'97*. 1997. ACM. pp. 155-156.
Keywords: *Tablet, multimodal input, modeling, 3D user interfaces, pen input*
Annotations: Describes an application which uses a tablet, 6DOF direct input and multimodal input, for rapid scene creation. The user can draw on the tablet, tracked in 3D using magnetic sensor and the 3D objects would "pop out" from the tablet, can be picked up and manipulated in virtual space.
 14. Bliss, J., Tidwell, P., Guest, M., The effectiveness of virtual reality for administering spatial navigation training to firefighters. *Presence: Teleoperators and Virtual Environments*, 1997. 6(1): pp. 73-86.
Keywords: *spatial orientation, training, knowledge transfer, wayfinding, navigation, VR*
Annotations: Article on knowledge transfer issues between virtual and real environments. Though domain specific, it provides several useful insights in what kinds of knowledge are transferred.
 15. Bolt, R., "Put-that-there": voice and gesture at the graphics interface. *Proceedings of SIGGRAPH'80*. 1980. ACM. pp. 262-270.
Keywords: *pointing, 3D interaction, interaction technique, multimodal interaction*
Annotations: 1. The work described involves the user commanding simple shapes about a large-screen graphics display surface. Because voice can be augmented with simultaneous pointing, the free usage of pronouns becomes possible, with a corresponding gain in naturalness and economy of expression. Conversely, gesture aided by voice gains precision in its power to reference. 2. One of the first papers that describes interface that involved spatial interaction. The paper features using of magnetic sensor, which was a novelty in those days, for selecting objects and moving them with voice commands.
 16. Bolter, J., Hodges, L., Meyer, T., Nichols, A., Integrating perceptual and symbolic information in VR. *IEEE Computer Graphics & Applications*, 1995. 15(4): pp. 8-11.
Keywords: *VR, menus, 3D interface, symbolic communication in VR, virtual text*
Annotations: This paper argues that user interfaces for VR should provide means to present and manipulate symbolic, textual information. A number of tasks where textual information is be needed are discussed: menu selection, presentation of numerical/statistical information, and presentation of narrative information, i.e. annotations.
 17. Bordegoni, M., Gesture Interaction in a 3D User Interface. GMD, Darmstadt: *Technical Report ERCIM-93-R019*. 1993.
Keywords: *Gesture interaction, manipulation, navigation, feedback, 3D interaction, virtual reality, glove input, multimodal interaction*
Annotations: Report on gesture interaction issues, describing a dynamic gesture language, needed feedback, a framework for a gesture system and several examples of gesture interaction in 3D user interfaces.
 18. Bowman, D., Davis, E., Badre, A., Hodges, L., Maintaining Spatial Orientation during Travel in an Immersive Virtual Environment. *Presence: Teleoperators and Virtual Environments*, 1999. 8(6): pp. 618-631.
Keywords: *taxonomy, navigation, spatial awareness, 3D maps, route-planning technique, VR, 3D user interfaces*
Annotations: This paper discusses a new taxonomy for virtual travel techniques and runs an experiment comparing three common travel metaphors. The experiment tests subjects' ability to remember the spatial relationship of an

object to the user's location after traveling through a virtual maze.

19. Bowman, D., Hodges, L., An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. *Proceedings of Symposium on Interactive 3D Graphics*. 1997. ACM. pp. 35-38.
Keywords: 3D interaction techniques, manipulation, virtual reality, selection, evaluation
Annotations: User study of simple manipulation techniques.
20. Bowman, D., Hodges, L., Formalizing the Design, Evaluation, and Application of Interaction Techniques for Immersive Virtual Environments. *The Journal of Visual Languages and Computing*, 1999. 10(1): pp. 37-53.
Keywords: 3D interaction, interaction techniques, taxonomy, testbed evaluation, manipulation, navigation, selection, user tasks, VR
Annotations: This article presents a methodology for designing, testing, and applying 3D interaction techniques in virtual environments. Taxonomies of techniques for travel, selection, and manipulation are discussed. The concepts of guided design and testbed evaluation are presented and examples are given. This is a summary of the methodology used in Bowman's dissertation.
21. Bowman, D., Johnson, D., Hodges, L., Testbed Evaluation of VE Interaction Techniques. *Proceedings of VRST'99*. 1999. ACM. pp. 26-33.
Keywords: formal evaluation, interaction techniques, manipulation, VR
Annotations: Testbed evaluation is a type of experimentation which attempts to obtain richer results by considering multiple independent and dependent variables. Here, two experiments and their results are described which test the performance and usability of techniques for the tasks of travel and selection/manipulation.
22. Bowman, D., Wineman, J., Hodges, L., Allison, D., Designing Animal Habitats Within an Immersive VE. *IEEE Computer Graphics & Applications*, 1998. 18(5): pp. 9-13.
Keywords: immersive design, pen and tablet metaphor, virtual manipulation, 3D interaction, immersive virtual reality, HMD
Annotations: The Virtual Habitat is an immersive VE which allows architects to redesign an animal habitat. This requires some complex and well-integrated interaction techniques for user travel and object manipulation.
23. Bowman, D.A., Koller, D., Hodges, L.F., Travel in immersive virtual environments: an evaluation of viewpoint motion control techniques. *Proceedings of IEEE VRAIS'96*. 1997. pp. 45-52.
Keywords: virtual reality, VR, 3D user interface, 3D interaction, navigation, viewpoint control, user study, experiments, interaction techniques
Annotations: Three formal experiments comparing common travel techniques
24. Britton, E., Lipscomb, J., Pique, M., Making nested rotations convenient for the user. *Proceedings of SIGGRAPH'78*. 1978. ACM. pp. 222-227.
Keywords: 3D manipulation, 3D interaction, input devices, 3D user interfaces, interactive rotations
Annotations: The early work which investigated the modes of interactive rotations using 6DOF input devices. The authors coin the term "kinaesthetic correspondence" a principle for 3D interface design which postulates that manipulated 3D object (which they called "subimage") should move in the same direction as user's hand.
25. Brooks, F., Grasping Reality Through Illusion: Interactive Graphics Serving Science. *Proceedings of CHI'88*. 1988. ACM. pp. 1-11.
Keywords: 3d interaction, framework, human factors
Annotations: 1. A very good paper with many useful insights on varying topics in 3D interaction / virtual reality. Includes his "shells-of-certainty" model for user interface research. 2. Good review of the various issues in VR and interactive 3D computer graphics up to 1988. However, many issues are still valid and probably will be valid for a while.
26. Brooks, F., Ouh-Young, J., Batter, J., Kilpatrick, P., Project GROPE- Haptic Displays for Scientific Visualization. In *SIGGRAPH'90*. 1990. ACM. pp. 177-185.
Keywords: haptics, display, visualization, virtual reality, manipulation
Annotations: Describes long-term research effort into haptic ("pertaining to sensations such as touch, temperature, pressure, etc. mediated by skin, muscle, tendon, or joint") displays for molecular docking. Interesting as an example of how to develop a system for real users. Haptic displays are of limited application, but when they are applicable, a performance increase of approximately 2x is measured over pure visual stimuli. Some interesting results on 3D/6D manipulation: * Users of an imperfect-perception visual system tend to decompose three-dimensional positioning tasks into several separate sub-tasks, each of lower dimensionality * Even in real space, subjects usually decompose 6D docking tasks into 3D positioning alternating with 3D rotations. More than 2D motions are rarely observed in virtual space.
27. Bukowski, R., Sequin, C., Object Associations: A Simple and Practical Approach to Virtual 3D Manipulation. *Proceedings of Symposium on Interactive 3D Graphics*. 1995. ACM. pp. 131-138.
Keywords: constraints, smart objects, 3D manipulation
Annotations: Presents a framework within which objects can be given intelligence about their proper positions and orientations within a 3D space to aid in manipulation
28. Burdea, G., *Force and Touch Feedback for Virtual Reality*. 1996: Wiley Interscience. pp. .
Keywords: haptic sensing, actuators, tactile feedback, physical modeling, force feedback, display devices
Annotations: Provides a comprehensive introduction and reference on force-feedback.
29. Butterworth, J., Davidson, A., Hench, S., Olano, T., 3DM: a three dimensional modeler using a head-mounted display. *Proceedings of Symposium on Interactive 3D Graphics*. 1992. ACM. pp. 135-138.
Keywords: 3D interaction, 3D interface toolkit, direct manipulation, interaction techniques, metaphor, navigating, virtual reality, manipulation, input devices, modeler
Annotations: 1. This article belongs to the roots of VR aided modelers and exploration of input devices. 2. The pioneering paper that introduced many of the most basic interaction techniques and ideas for 3D interfaces and virtual reality. 3. Describes a 3D CAD system for use in a HMD. Has support for multiple navigation models: User "growing" and "shrinking" to allow work at multiple levels of detail; Walking (only within tracker range); Flying;

- Grabbing the world (dragging & rotating). Uses rubber banding and predictive highlighting (e.g. gravity and plane/grid snapping) to aid in object selection. Simultaneous translation and rotation is helpful because it "concentrates more functionality into each operation" (thus saving time by requiring fewer total operations).
30. Buxton, W., Touch, Gesture, and Marking. In *Readings in Human-Computer Interaction: Toward the Year 2000*, R. Baecker, et al., Editors. 1995, Morgan Kaufmann.
Keywords: *taxonomy, chunking and phrasing, input devices, survey*
Annotations: 1. An excellent overview including device capabilities, taxonomy of input devices, chunking and phrasing, marking, gestures, and two handed input. Lots of good references to key papers in the area. 2. This is a survey which can be very useful for everybody who is working in 3D user interface design: indeed great many issues we face in designing 3D interfaces are similar or the same to those in 2D interfaces
31. Buxton, W., Myers, B., A Study in Two-handed Input. *Proceedings of CHI'86*. 1986. ACM. pp. 321-326.
Keywords: *two-handed input*
Annotations: 1. Valuable and classical survey on two-handed input 2. This is the classical work on 2 handed input. Although it does not directly relate to 3D user interfaces, it is probably a must read for everybody who plans to do work on 2 handed input in 3D.
32. Chen, M., Mountford, S.J., Sellen, A., A Study in Interactive 3-D Rotation Using 2-D Control Devices. *Proceedings of SIGGRAPH'88*. 1988. ACM. pp. 121-129.
Keywords: *object rotation, 2D input, desktop 3D interfaces, mouse, user studies, experimental evaluation*
Annotations: Chen studies four methods for using 2D input to rotate 3D objects: 1) Graphical sliders: A simple arrangement of horizontal sliders, one each for x, y, and z rotations. 2) Overlapping sliders: Uses vertical/horizontal mouse movement to control x and y rotations, while circular movement means z rotation. 3) Continuous XY + Z: 4) Virtual Sphere: Chen's user study indicated that the Virtual Sphere technique achieved the best results. He also compared the Virtual Sphere with a similar technique developed by Evans et al. [Evans 81]; no significant difference was found in mean time to complete simple or complex rotations, but users preferred the Virtual Sphere controller. The paper includes an appendix which describes the implementation of the virtual sphere in detail.
33. Chung, J.C., A comparison of Head-tracked and Non-head-tracked Steering Modes in the Targeting of Radiotherapy Treatment Beams. *Proceedings of Symposium on Interactive 3D Graphics*. 1992. ACM. pp. 193-196.
Keywords: *steering, tracking*
Annotations: This study compares four head-tracked and three non-head-tracked modes for changing position and orientation in the virtual world. Taken as a whole, head-tracked and non-head-tracked modes "differed very little". The test model was an abstract model consisting of colored spheres and a central target region. The user tried to find the best beam path to the target, which was defined as the beam path with minimum intersection of the beam and the spheres. All interaction modes were displayed on a HMD (N=14 subjects).
34. Conner, B., Snibbe, S., Herndon, K., Robbins, D., Zelenik, R., et al., Three-dimensional widgets. *Proceedings of Interactive 3D graphics Symposium*. 1992. pp. 183-188.
Keywords: *3D interfaces, 3D widgets, interaction techniques, 3D interaction*
Annotations: The original paper that introduced 3D widgets for 3D interfaces as a first-class objects in the virtual environments.
35. Coquillart, S., Wesche, G., The virtual palette and the virtual remote control, a device and an interaction paradigm for the responsive workbench. *Proceedings of Virtual Reality'99*. 1999. IEEE. pp. 213-216.
Keywords: *transparent tablet, responsive workbench, magic lenses, two-handed interaction, props*
Annotations: The authors introduce a prop-like device, the Virtual Palette, a transparent tablet with a handle, tracked using magnetic sensor, which is used to interact with the responsive workbench. The user looks at the responsive workbench through the tablet, and the image on the workbench is registered with the tablet so that it appears to be on the surface of the tablet. The user can interact with the image by touching it on the physical tablet with a pencil tracked with magnetic sensor. Authors also describe the Virtual Remote Control Panel, a two-handed interaction technique, based on the Virtual Palette to control applications.
36. Cruz-Niera, C., Sandin, D., Defanti, T., Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE. *Proceedings of SIGGRAPH'93*. 1993. ACM. pp. 135-142.
Keywords: *virtual reality, VR, stereoscopic display, output devices, head tracking, projection paradigms, real-time manipulation, immersion*
Annotations: Describes the design and implementation of the Cave Automatic Virtual Environment, a 4 wall projection-based VR display system. The paper also goes into some detail on off-axis projections techniques.
37. Cugini, J., Laskowski, S., Sebrecchts, M., Design of 3D Visualization of Search Results: Evolution and Evaluation. *Proceedings of 12th Annual International Symposium: Electronic Imaging 2000: Visual Data Exploration and Analysis*. 2000. IST/SPIE. pp. 198-210.
Keywords: *3D user interfaces, visualization, information retrieval*
Annotations: The paper discusses the evolution of the NIST Information Retrieval Visualization Engine (NIRVE). This prototype employs modern interactive visualization techniques to provide easier access to a set of documents resulting from a query to a search engine. The motivation and evaluation of several design features, such as keyword to concept mapping, explicit clustering, the use of 3-D vs. 2-D, and the relationship of visualization to logical structure are described.
38. Darken, R., Hands-off interaction with menus in virtual space. *Proceedings of Stereoscopic Displays and Virtual Reality Systems*. 1994. SPIE. pp. 365-371.
Keywords: *Menus, system control, visibility, readability*
Annotations: The author describes issues relating to the usage of menus in 3D virtual environments. A special focus was at visibility of menus in 3D and readability of fonts in VR. The author suggests a number of guidelines and principles on menu placement in VR.

39. Darken, R., Allard, T., Achille, L., Spatial Orientation and Wayfinding in Large-Scale Virtual Spaces: An Introduction. *Presence*, 1998. 7(2): pp. 101-107.
Keywords: *Wayfinding, navigation, spatial cognition, search methods*
Annotations: Excellent introduction to wayfinding issues
40. Darken, R., Cevik, H., Map usage in virtual environments. *Proceedings of VR'99*. 1999. IEEE. pp. 133-140.
Keywords: *wayfinding, maps, cues, map orientation, navigation, 3D interaction, VR*
Annotations: This article investigates how a map should be used during navigation in a virtual environment. Particularly, it focuses especially on effect of forward-up and north-up maps orientation on user performance during wayfinding. The authors found correlation between map orientation, reference frame and search task.
41. Darken, R., Cevik, H., Map Usage in Virtual Environments: Orientation Issues. *Proceedings of Virtual Reality '99*. 1999. IEEE. pp. 133-140.
Keywords: *Wayfinding, maps, orientation*
Annotations: This article describes issues involved in the usage of maps into virtual environments. The authors conclude with guidelines for the usage of maps: via several tests, it was found out, that with egocentric search tasks, a forward-up map is preferable, whereas with exocentric search tasks, a north up performs best.
42. Deering, M., High Resolution Virtual Reality. *Computer Graphics*, 1992. 26(2): pp. 195-202.
Keywords: *desktop vr, head tracking*
Annotations: Talks about a desktop VR system which allows the user to work with a 3D tracker in a volume stereoscopically projected in front of the monitor. Good description of the math for head tracking. Also talks about taking into account the user's actual eye and distortions caused by the monitor glass.
43. Deering, M., The HoloSketch VR Sketching System. *Communications of the ACM*, 1996. 39(5): pp. 54-61.
Keywords: *3D modeling, output devices, input devices, 3D sketching*
Annotations: Describes how 2D sketching can be ported to VEs and reports on several issues, like menu systems.
44. Doellner, J., Hinrichs, K., Interactive, Animated 3D Widgets. *Proceedings of Computer Graphics International '98*. 1998. IEEE. pp. 278-286.
Keywords: *3D widgets, behavior, visual language, system control*
Annotations: This paper describes an object-oriented architecture for interactive, animated 3D widgets. Two types of directed acyclic graphs (geometry graphs and behavior graphs) on which operations are performed through high-level interfaces. A visual language for 3D widgets allows the developer to interactively construct 3D applications.
45. Draper, M., Exploring the Influence of a Virtual Body on Spatial Awareness, in *Department of Engineering*. 1995. University of Washington: Seattle.
Keywords: *Spatial awareness, wayfinding, virtual body, distance estimations*
Annotations: Master thesis exploring a broad number of factors relating to spatial awareness, especially those resulting from using a virtual body. Reports that no specific positive effects of using a virtual body on users spatial awareness were found.
46. Elvins, T., Nadeau, D., Kirsh, D., Worldlets - 3D Thumbnails for Wayfinding in Virtual Environments. *Proceedings of UIST'97*. 1997. pp. 21-30.
Keywords: *virtual reality, navigation, viewpoint control, 3D interaction techniques, wayfinding, landmark knowledge, WIM*
Annotations: Interesting article which describes the usage of so called Worldlets, 3D thumbnails representing a landmark in a virtual environment, to support wayfinding.
47. Encarnacao, L., Bimber, O., Schmalstieg, D., Chandler, S., A Translucent Sketchpad for the Virtual Table. *Computer Graphics Forum*, 1999. 18(3): pp. 277-286.
Keywords: *Two-handed interaction, virtual workbench, gestural interaction, props, see-through tools*
Annotations: A discussion of a two-handed pen and pad style interaction method for a virtual workbench. A detailed explanation of a 2D pen gesture recognition technique is given.
48. Encarnacao, L., Fechter, J., Grunert, T., Strasser, W., A Platform for User-Tailored Interaction Development in 2D, 3D and VR. *Computer Graphics Forum*, 1996. 15(3): pp. 432-441.
Keywords: *User interface design, interaction objects, virtual interfaces*
Annotations: A platform for the development, integration and user-centered evaluation of interaction techniques. A message-passing layer communicates with a series of application objects representing 2D, 3D and VR.
49. Evans, K.B., Tanner, P.P., Wein, M., Tablet-based Valuators that Provide One, Two, or Three Degrees of Freedom. *Computer Graphics*, 1981. 15(3): pp. 91-97.
Keywords: *positioning techniques, desktop 3D user interfaces, stylus, rotation*
Annotations: Describes various ways of mapping stylus motion to valuators. One of his 3DoF techniques is similar to the Virtual Sphere; Chen compares it to the Virtual Sphere in his paper [Chen 1988]. Evans also discusses an automatic vernier motion (fine positioning) technique.
50. Feiner, S., MacIntyre, B., Knowledge-Based Augmented Reality. *Communications of the ACM*, 1993. 36(7): pp. 53-61.
Keywords: *augmented reality, AR, HMD, see-through*
Annotations: Describes a system which employs a see-through head mounted display (augmented reality) and projects wireframe graphics onto objects in the real world. An example given is an application which overlays a laser printer with wireframe information to help the user perform maintenance tasks. The head mount is constructed using a Private Eye.
51. Feiner, S., MacIntyre, B., Haupt, M., Solomon, E., Windows on the world: 2d windows for 3d augmented reality. *Proceedings of UIST'93*. 1993. ACM. pp. 145-155.
Keywords: *Menus, system control, augmented reality, 3D user interfaces*
Annotations: The authors explore three kinds of widgets, which were 2D windows overlapped on physical world in their augmented environment application: surround-fixed windows, display-fixed windows, world-fixed windows. The authors discuss widgets placement and frame rate issues. Although the paper primarily deals with the aug-

mented reality applications, conclusions and results can be used in designing any 3D interface.

52. Fisher, S., McGreevy, M., Humphries, J., Robinett, W., Virtual Environment Display System. *Proceedings of Workshop on Interactive 3D Graphics*. 1986. ACM.
Keywords: *display, two-handed interaction*
Annotations: An excellent piece of early virtual reality research. NASA Telepresence research. Not mentioned in the text, but clearly the authors envisioned two-handed manipulation (along with voice input and 3D localized sound).
53. Foley, D., Wallace, V., Chan, V., The human factors of computer graphics interaction techniques. *IEEE Computer Graphics & Applications*, 1984(4): pp. 13-48.
Annotations: This is one of the most fundamental papers on interaction techniques for graphical user interfaces. This paper was probably one of the first attempts to break down complex interaction sequences into several basic interaction tasks and propose that each elementary interaction task is accomplished by the means of the certain interaction techniques. The paper surveys many of the interaction techniques and while most of them are 2D some 3D techniques are also briefly discussed. Parts of this paper have been included in the well known textbook on computer graphics by Foley, van Dam and others.
54. Forsberg, A., Herndon, K., Zeleznik, R., Aperture based selection for immersive virtual environment. *Proceedings of UIST'96*. 1996. ACM. pp. 95-96.
Keywords: *VR, virtual reality, 3D interaction techniques, direct manipulation, selection, 3D interaction, pointing*
Annotations: Describes an interaction technique for selecting objects in immersive VR. The technique is an extension of the flash-light technique (see Liang, 1994): it allows to interactively control the size of the conic selection volume which, in turn, allows easier disambiguation of target objects.
55. Forsberg, A., LaViola, J., Markosian, M., Zeleznik, R., Seamless Interaction In Virtual Reality. *IEEE Computer Graphics and Applications*, 1997. 17(6): pp. 6-9.
Keywords: *seamless integration, ErgoSketch, 2D/3D input, VR, responsive workbench, interaction techniques, 3D interfaces*
Annotations: This paper talks about the importance of seamlessly combining 2D and 3D interaction techniques and discusses when each interaction metaphor is appropriate in the context of 3D modeling.
56. Forsberg, A., LaViola, J., Zeleznik, R., ErgoDesk: A Framework For Two and Three Dimensional Interaction at the ActiveDesk. *Proceedings of Second International Immersive Projection Technology Workshop*. 1998.
Keywords: *ActiveDesk, sketch, 3D interaction, 3d modeling, two-handed input*
Annotations: This paper presents a hardware and software framework for combining 2D and 3D interaction in projection-based virtual reality.
57. Froehlich, B., Plate, J., The Cubic Mouse, A New Device for Three-Dimensional Input. *Proceedings of CHI*. 2000. ACM.
Keywords: *Input device, props, visualisation, interaction, cutting and slicing planes*
Annotations: The authors describe a new device for three-dimensional input. The device is built up from a cube-shaped box with a tracker, and three rods running through the axes of the cube. The device allows very fine interaction with several described applications, and shows good evaluation results from a performed user study.
58. Galyean, T., Guided navigation of virtual environments. *Proceedings of Symposium on Interactive 3D Graphics*. 1995. ACM. pp. 103-104.
Keywords: *VR, navigation, 3D interface, viewpoint control*
Annotations: Interesting technique for navigation in VR where the user is guided along the path in the environment and yet has some degree of freedom to explore it. The technique is based on the "The River Analogy" metaphor, where the user is like a boat floating down a river and pulled by the stream and just lie a bout he or she can diverge from the strait path to look around. The technique can be very useful in designing interfaces for narrative, story telling VR environments.
59. Gobbetti, E., Balaguer, J., VB2 : A Framework for Interaction in Synthetic Worlds. *Proceedings of UIST*. 1993. ACM. pp. 167-178.
Keywords: *User interface design, 3D virtual tools, gestural input, 3D user interaction*
Annotations: The paper describes the VB2 architecture for the construction of three-dimensional interactive applications. The authors deal with virtual tools, constraints, gestural input and direct manipulation. An example application domain, animation, is used to illuminate the topics.
60. Goble, J., Hinckley, K., Pausch, R., Snell, J., Kassell, N., Two-Handed Spatial Interface Tools for Neurosurgical Planning. *Computer*, 1995. 28(7): pp. 20-26.
Keywords: *props, 3D input, domain-specific interaction, two-handed interaction*
Annotations: Presents the Netra system, important for its use of real-world props to aid in 3D manipulation, and the concept of clutching during 3D manipulation.
61. Goesele, M., Stuerzlinger, W., Semantic Constraints for Scene Manipulation. *Proceedings of Spring Conference in Computer Graphics '99*. 1999. pp. 140-146.
Keywords: *Semantics, manipulation, 3D user interface*
Annotations: The system uses semantic/pragmatic constraints to simplify the 3D user interface for a Virtual Reality system. It builds upon Bukowski and Sequin's work.
62. Grissom, S., Perlman, G., StEP(3D): A standardized evaluation plan for three-dimensional interaction techniques. *International Journal of Human-Computer Studies*, 1995. 43(1): pp. 15-41.
Keywords: *testbed, 3D user interface, manipulation, interaction techniques*
Annotations: The testbed for evaluation of 3D interaction techniques. For another example of experimental testbed see [Poupyrev, et al. 1997]
63. Hand, C., A Survey of 3D Interaction Techniques. *Computer Graphics Forum*, 1997. 16(5): pp. 269-281.
Keywords: *VR, 3D, survey, interaction technique, manipulation, navigation, viewpoint control, feedback, widgets*
Annotations: Interesting and pretty extensive survey of interaction techniques for desktop and immersive VR and issues related to their development.

64. Harmon, R., Patterson, W., Ribarsky, W., Bolter, J., The virtual annotation system. *Proceedings of VRAIS'96*. 1996. IEEE. pp. 239-245.
Keywords: *virtual annotation system, annotation tools, voice annotations, icon, architectural walkthrough, virtual reality, 3D interaction*
Annotations: 1. The paper presents a set of voice annotation tools that can be placed in a variety of VR applications. These tools offer a set of capabilities for inserting, iconizing, playing back and organizing voice annotations in a virtual space. 2. See also Verlinden, Bolter, et al. 1993 where this idea was first introduced.
65. Harris, L., Jenkin, M., Zikovitz, D., Vestibular cues and virtual environments: choosing the magnitude of the vestibular cue. *Proceedings of VR'99*. 1999. IEEE. pp. 229-236.
Keywords: *Vestibular cues, receptors, real-motion cues, cart, perception, VR*
Annotations: This article reports the correlation between visual input and real-motion cues. Reported is, that a virtual reality system designer should supply four times as much visual motion as vestibular motion to obtain accuracy during passive motion.
66. Henry, D., Furness, T., Spatial perception in virtual environments: evaluating an architectural application. *Proceedings of VRAIS'93*. 1993. IEEE. pp. 33-40.
Keywords: *spatial representation, perception, wayfinding, architectural space, VR, navigation*
Annotations: 1. Early report on factors influencing the effectiveness of representation of architectural space within virtual environments. Henry reports distance estimation and orientation biases between virtual environments and real world environments. 2. The article compared how users perceive real and virtual environments. The author designed a virtual environment which replicates exactly a physical environment and aligned them "on top" of each other. He founded and described differences in user perception of both environments and suggest reasons while this differences occur(e.g. limited vertical field of view of HMD)
67. Herndon, K., Meyer, T., 3D widgets for exploratory scientific visualization. *Proceedings of UIST'94*. 1994. ACM. pp. 69-70.
Keywords: *Widgets, system control, interactive shadows*
Annotations: The paper describes several 3D widgets for scientific data exploration and further exploration of previous work on interactive shadows. The authors also describe several design issues related to geometry, dimensionality and user feedback.
68. Herndon, K., van Dam, A., Gleicher, M., The challenges of 3D interaction: a CHI'94 workshop. *SIGCHI Bulletin*, 1994. 26(4); pp. 36-43.
Keywords: *3D interaction, survey*
Annotations: 1. A report on a CHI workshop listing most of the important challenges and research direction in 3D interaction. 2. Summarizes discussions held at the CHI'94 Workshop on 3D interaction. Covers a wide range of topics, including applications of 3D graphics, psychology and perception issues, state of the art work, and future research directions. Includes an excellent bibliography.
69. Herndon, K.P., Zeleznik, R.C., Robbins, D.C., Conner, D.B., Snibbe, S.S., *et al.*, Interactive shadows. *Proceedings of UIST'92*. 1992. ACM. pp. 1-6.
Keywords: *interactive shadows, spatial relationships, 3D interaction techniques, manipulation techniques, 3D widgets, shadow widgets, 3D user interfaces, desktop 3D interaction*
Annotations: Paper present a set of 3D widgets called "shadows" that provide perceptual cues about the spatial relationships between objects, and also provide a direct manipulation interface to position objects. Unlike some other 3D widgets, they do not obscure the objects they control.
70. Hinckley, K., Pausch, R., Goble, J., Kassell, N., A survey of design issues in spatial input. *Proceedings of UIST '94*. 1994. ACM. pp. 213-22.
Keywords: *spatial input, design issues survey, 3D interaction, 3D interfaces, virtual environment, VR, two-handed interaction, feedback, physical constraints, head tracking, interaction techniques.*
Annotations: 1. A survey of design issues for developing effective free-space three-dimensional (3D) user interfaces based upon authors previous work in 3D interaction, our experience in developing free-space interfaces, and informal observations of test users. Can serve as a guide to researchers or systems builders. 2. A practical and useful set of principles to follow when 3D input is given to a computer system.
71. Hinckley, K., Pausch, R., Proffitt, D., Patten, J., Kassell, N., Cooperative bimanual action. *Proceedings of CHI'97*. 1997. ACM. pp. 27-34.
Keywords: *manipulation, 3D interfaces, bi-manual interaction, two-handed interaction, experimental evaluation*
Annotations: Paper presents experiments on two-handed manipulation. The paper concentrates on division of labor between two hands when performing a single task, i.e. cooperative two-handed manipulation. Experiments suggest that left hand defines a spatial frame of reference for the right hand for complex manipulation actions. The contribution of hands is, therefore, asymmetric.
72. Hinckley, K., Tullio, J., Pausch, R., Proffitt, D., Kassell, N., Usability Analysis of 3D Rotation Techniques. *Proceedings of UIST'97*. 1997. pp. 1-10.
Keywords: *Arcball, Virtual Sphere, 6DOF input devices, usability, manipulation, 3D interaction technique, interactive rotation, experimental studies*
Annotations: 1. Good report on user study of 3D rotation using mouse-driven Virtual Sphere and ARCBALL techniques, as well as 6DOF input devices. 2. The main result of this paper is that 6DOF devices do allow for better user performance in rotation task without sacrificing accuracy. Physical form of device has also been investigated, however, no performance difference have been found for devices with different shapes. The shape of device, however, did influence the user acceptance and subjective rating of device.
73. Hinkley, K., Pausch, R., Goble, J., Kassell, N., Passive Real-World Interface Props for Neurosurgical Visualization. *Proceedings of CHI'94*. 1994. ACM. pp. 452-458.
Keywords: *3D interaction, gesture input, two-handed interaction, haptic input, neurosurgery, visualization, 3D interfaces*
Annotations: Classical article on usage of props. Excellent review of possibilities and problems
74. Hoffman, H., Physically touching virtual objects using tactile augmentation enhances the realism of virtual envi-

- ronments. *Proceedings of VRAIS'98*. 1998. IEEE. pp. 59-63.
Keywords: *VR, virtual reality, realism, haptic and tactile feedback, physical props, experimental study*
Annotations: Reports experimental study which empirically demonstrated that adding tactile augmentation, or simply props, can increase realism of the virtual environment. Argues a value of adding props in 3D interface design for VR. See also [Hinckley, 1994]
75. Houde, S., Iterative Design of an Interface for Easy 3-D Direct Manipulation. *Proceedings of CHI'92*. 1992. ACM. pp. 135-142.
Keywords: *3-D manipulation, bounding box, direct manipulation, hand gestures, handle box, iterative design, narrative handles, space planning*
Annotations: Describes a system with handles on object for 3D manipulation; hand-shaped cursors suggest type of manipulation being performed. The system must switch modes when going between translations and rotations.
76. Howard, I., Spatial vision within egocentric and exocentric frames of reference. In *Pictorial communication in virtual and real environments*, S. Ellis, et al., Editors. 1991, Tayler and Francis Ltd.: London. pp. 338-357.
Keywords: *frames of reference, perception, wayfinding, vection, VR*
Annotations: An excellent survey on how egocentric and exocentric reference frames are built, and how they function. With examples gained from tests, Howard also illuminates several related factors like vection.
77. Hultquits, J., A Virtual Trackball. In *Graphics Gems I*. 1990, Academic Press. pp. 462-463.
Keywords: *3D rotations, desktop interface, mouse, trackball*
Annotations: Technique to control 3D rotations using a mouse. Mouse movements, sampled repeatedly, are used to compute instantaneous rotation axis of the 3D object. See also [Shoemake, 92, Chen, 88, Hinckley'97]
78. Ingram, R., Bowers, J., Benford, S., Building virtual cities: applying urban planning principles to the design of virtual environments. *Proceedings of VRST'96*. 1996. ACM.
Keywords: *wayfinding, structuring, perception, Lynch, urban planning, VR*
Annotations: An interesting article on how Lynch' Image of the City can be applied for the design of virtual environments.
79. Iwata, H., Fujii, T., Virtual Perambulator: A Novel Interface Device for Locomotion in Virtual Environment. *Proceedings of VRAIS'96*. 1996. IEEE. pp. 60-65.
Keywords: *travel, locomotion, natural interaction, walking technique, 3D interaction, VR, virtual reality*
Annotations: One of the many systems that attempts to simulate walking by having the user walk in place. Requires trackers only, and no expensive hardware
80. Jacob, R., Deligiannidis, L., Morrison, A Software Model and Specification Language for Non-WIMP User Interfaces. *Transactions on Computer-Human Interaction*, 1999. 6(1): pp. 1-46.
Keywords: *3D user interfaces, interaction techniques, non-WIMP interface, specification language*
Annotations: A software model and language for describing and programming interaction in non-WIMP user interfaces is presented. The model combines a data-flow or constraint-like component for the continuous relationships with an event-based component for discrete interactions, which can enable or disable individual continuous relationships. The description of the PMIW user interface management system demonstrates the approach. The main goal is to provide a model and language that captures the formal structure of non-WIMP interactions in the way that various previous techniques have captured command-based, textual and event-based styles.
81. Jacob, R., Sibert, L., The Perceptual Structure of Multidimensional Input Device Selection. *Proceedings of CHI'92*. 1992. ACM. pp. 211-218.
Keywords: *polhemus tracker, gesture input, input devices, integrality, interaction techniques, perceptual space, separability*
Annotations: This study addresses the question: "What is a three-dimensional tracker good for?" The authors hypothesize that "the structure of the perceptual space of an interaction task should mirror that of the control space of its input device." Thus, a 3D tracker would be good for a task which involves the selection of three related ("integral") dimensions, but would be less effective for unrelated ("separable") dimensions. The study had users perform two interaction tasks with both a Polhemus and a mouse. One task involved setting three integral parameters (x, y location and size of a rectangle), while the other involved separable parameters (x, y location and color of a rectangle). The data collected suggested that matching the integrality/separability of the device to the task yields the best user performance. Neither the Polhemus or the mouse was uniformly superior; each device performed best when it was correctly mapped to "the perceptual structure of the task space".
82. Jacoby, R., Ellis, S., Using Virtual Menus in a Virtual Environment. *Proceedings of Visual Data Interpretation, 1668*. 1992. SPIE. pp. 39-48.
Keywords: *3D menus, commands, 3D interaction*
Annotations: One of the first systems to use virtual pull-down menus in a VE
83. Kaufman, A., Yagel, R., Tools for Interaction in Three Dimensions. *Proceedings of 3rd International Conference on Human-Computer Interaction*. 1989. pp. 468-475.
Keywords: *display system, jack, projection methods, desktop 3D user interface*
Annotations: This paper contains the most comprehensive description of the 3D user interface for Kaufman's CUBE workstation. Cube has viewing windows which employ a "combination look" for object rendering: drawings are superimposed on shaded images to capitalize on the advantages of each type of look. A separate window ("World Space") allows the user to specify the eye point, the direction of projection, the projection surface, the light sources (3), etc. The world view can be merged with the view window on sufficiently fast machines. A "full jack" or a jack with shadows on each wall is used to relate position information. The paper advocates having anchors in each objects to help with positioning; this is mostly useful in geometric objects which have been created in the environment (to define volumes of interest or surgical implants). A gravity mechanism is used to assist motion during object picking and parameter specification.
84. Kessler, G., A Framework for Interactors in Immersive Virtual Environments. *Proceedings of VR'99*. 1999. IEEE. pp. 190-197.

- Keywords:** *Interaction techniques, framework, user interface*
- Annotations:** SVIFT, the Simple Virtual Interactor Framework and Toolkit, is presented to meet the interaction needs of immersive VE applications. SVIFT allows for the design and implementation of various interaction techniques that can be easily incorporated into many VE applications and combined with other interaction techniques. Differences between desktop and immersive environment interaction are also discussed.
85. Koller, D., Mine, M., Hudson, S., Head-tracked orbital viewing: An interaction technique for immersive Virtual Environments. *Proceedings of UIST'96*. 1996. ACM. pp. 81-82.
Keywords: *object viewing, head-based manipulation, viewpoint control, 3D interaction, virtual reality*
Annotations: This technique maps the user's head motion to the view the user receives of a virtual object - look up to see the bottom of the object, look down to see the top and etc.
86. Krueger, M., Gionfriddo, T., Hinrichsen, K., VIDEOPLACE - An Artificial Reality. *Proceedings of CHI'85*. 1985. ACM. pp. 35-40.
Keywords: *artificial reality, video-based tracking, gestures, projection, gesture input*
Annotations: In VIDEOPLACE, one of the most compelling examples is using both hands to edit a B-spline curve: you can use index finger & thumb of each hand to simultaneously manipulate 4 control points at once. Even though the system is over 10 years old, in many ways it offered much richer interaction than present day technologies.
87. Latta, J.N., Oberg, D.J., A conceptual virtual reality model. *IEEE Computer Graphics & Applications*, 1994. 14(1): pp. 23-29.
Keywords: *3D interaction, VR, virtual reality, theory, conceptual models, perception, sensation.*
Annotations: Presents a conceptual model of VR. First, the definition of VR is suggested as a user interface to human perceptual and muscle system (listed) which objective is to place the user in an environment that is not normally or easily experienced. Several views of the VR systems are presented after that: human view and technical view. The human view is described basically from the point of system effectors, i.e. what user can feel, and system sensors, i.e. what user actions can be captured by the system. The technical view is a usual diagram of various components of the VR systems.
88. LaViola, J., Zeleznik, R., Flex and Pinch: A Case Study of Whole-Hand Input Design for Virtual Environment Interaction. *Proceedings of International Conference on Computer Graphics and Imaging'99*. 1999. IASTED. pp. 221-225.
Keywords: *3D graphics applications, conductive cloth, flex and pinch input, multimodal interaction, 3D user interfaces, VR, gesture and glove input, interaction techniques*
Annotations: This paper describes a hybrid input device that combines the continuous bend sensors from a data glove and the discrete contact sensors from the Fakespace Pinch glove. It describes improvements to a number of existing 3D interaction techniques.
89. LaViola, J.J., A survey of hand postures and gesture recognition techniques and technology. Brown University, Providence: *Technical Report CS-99-11*. 1999.
Keywords: *Glove interaction, gestures, postures, recognition techniques*
Annotations: An overview of gesture interaction technology for multimodal interaction with focus on VR and 3D UI. The survey reviews various gesture capturing and recognition technologies.
90. LeBlanc, A., Kalra P, Magnenat-Thalmann, N., Thalmann, D., Sculpting with the "Ball and Mouse" Metaphor. *Proceedings of Graphics Interface '91*. 1991. pp. 152-159.
Keywords: *two-handed interaction, modeling, spacemouse, 3D user interfaces, manipulation, desktop*
Annotations: Describes a two-handed 3D interface based on orienting object with spaceball in left hand (rotations only) and grabbing it with the mouse
91. Liang, J., Green, M., JDCAD: A highly interactive 3D modeling system. *Computer & Graphics*, 1994. 4(18): pp. 499-506.
Keywords: *geometric modeling, 3D interfaces, constraints, input devices, selection techniques.*
Annotations: 1. Good article on constrained (1DOF) menu systems and bat usage 2. The flash light technique that uses conic selection volume for 3D object selection has been introduced here. 3. Describes a Polhemus-based CAD system. The user hold the polhemus in front of the monitor and casts rays into the scene, rather than picking directly based on the position of the polhemus. This provides a nice metaphor for working at increased scale -- the user can zoom in on an object to see detail; since everything is done relative to the image on the monitor, a hand motion in real space now results in a small-scale motion in virtual space. A lot of interesting ideas.
92. Lindeman, R., Sibert, J., Hahn, J., Hand-held Windows: Towards Effective 2D Interaction in Immersive Virtual Environments. *Proceedings of VR'99*. 1999. IEEE. pp. 205-212.
Keywords: *2D widgets, pen and tablet interaction, two-handed interaction*
Annotations: The authors describe a testbed taking advantage of bimanual interaction, proprioception, and passive-haptic feedback to perform more precise manipulations in immersive virtual environments using 2D interaction techniques. They use a window registered with a tracked, physical surface to provide support for precise manipulation of interface widgets displayed in the virtual environment.
93. Liu, A., Stark, L., Hirose, M., Interaction of Visual Depth Cues and Viewing Parameters During Simulation Telemanipulation. *Proceedings of IEEE International Conference on Robotics and Automation*. 1991. pp. 2286-2291.
Keywords: *Telemanipulation, visual cues and parameters*
Annotations: User study. Tests the effectiveness of head motion parallax, but the motion was not under user control: the view simply oscillated under machine control. "Our experimental results do not provide strong evidence that relative depth cues affected tasks that required absolute depth information. The object rotation cue did not enhance task performance because it only provided information about the object's three dimensionality. Pseudo-head motion parallax as we implemented it, also did not no enhance performance, but if implemented under operator

- control, it might prove to be a more effective cue. The frustrum angle decreased completion time but had no effect on error."
94. Mackinlay, J., C., S., Robertson, G., Rapid controlled movement through a virtual 3D workspace. *Proceedings of SIGGRAPH'90*. 1990. ACM. pp. 171 - 176.
Keywords: *transitional motion, automated travel, interaction technique, 3D interaction*
Annotations: This technique uses the "slow-in, slow-out" movement favoured by animators in moving a user's viewpoint through a 3D space.
95. Mapes, D., Moshell, J., A Two-Handed Interface for Object Manipulation in Virtual Environments. *Presence: Teleoperators and Virtual Environments*, 1995. 4(4): pp. 403-416.
Keywords: *virtual workbench, pinch glove interaction, 3D manipulation, two-handed interaction, gestures, multimodal input, virtual reality, 3D interfaces*
Annotations: 1. This system, which evolved into an actual Multigen product, has interesting 2-handed techniques for viewpoint motion and object manipulation. 2. Polyshop system which has become SmartScenes product by Multigen.
96. Massimo, M., Sheridan, T., Roseborough, J., One Handed Tracking in Six Degrees of Freedom. *Proceedings of International Conference on Systems, Man and Cybernetics*. 1989. IEEE. pp. 498-503.
Keywords: *Tracking, spaceball, 6DOF, 3D user interfaces, experiments, user studies, devices*
Annotations: This article reports on user experiments for controlling 1, 3, and 6 degrees of freedom at a time for "pursuit" tracking tasks with a sensor ball (apparently identical to a spaceball) as an input device. It also tests the usage of velocity control vs. acceleration control. In all cases (1, 3, 6 DoF) z translations were the most difficult to control, and velocity input yielded better control than acceleration input. The use of shadows as depth cues did not help z translations.
97. McKenna, M., Interactive Viewpoint Control and Three-Dimensional Operations. *Proceedings of Symposium on Interactive 3D Graphics*. 1992. ACM. pp. 53-56.
Keywords: *Fish tank VR, tracking, viewpoint control, head tracking, desktop VR, 3D interfaces*
Annotations: Describes a "fish tank VR" system which changes the image on a standard 2D monitor based on head position. This allows perspective and motion parallax (monocular depth cues) without a HMD. An extension of this technique also tracks the monitor, allowing additional freedom (e.g. translation/rotation of the monitor). A Polhemus sensor is used to track the head; a stereoscopic version is described but not implemented.
98. McMillan, G., Eggeston, R., Anderson, T., Nonconventional controls. In *Handbook of human factors and ergonomics*, G. Salvendy, Editor. 1997, John Wiley and Sons: New York. pp. 729-771.
Keywords: *System control, speech, gestures, tracking*
Annotations: An excellent chapter on nonconventional control methods, like gestures, speech and gaze tracking. The authors also supply several user feedback guidelines. The material in this chapter would be especially beneficial to everybody who want to use these less conventional techniques in designing multimodal 3D user interfaces.
99. Mine, M., Virtual environment interaction techniques. UNC Chapel Hill CS Dept.: *Technical Report TR95-018*. 1995.
Keywords: *3D interaction. 3D user interface. 3D widgets, direct manipulation. interaction techniques. navigating, user interface. viewpoint control. VR.*
Annotations: 1. An overview of many of the most important early techniques for travel, selection, and manipulation in immersive VEs. 2. Good review, and must read for anybody who works with 3D interaction. However, since it was done in 1995 many newest and interesting ideas and techniques are not there.
100. Mine, M., ISAAC: A Meta-CAD System for Virtual Environments. *Computer-Aided Design*, 1997. 29(8): pp. 547-553.
Keywords: *immersive design, constraints, 3D manipulation, virtual menus*
Annotations: A system for describing the layout of 3D objects immersively. Uses several novel and well-constrained interaction techniques.
101. Mine, M., Brooks, F., Sequin, C., Moving objects in space: exploiting proprioception in virtual-environment interaction. *Proceedings of SIGGRAPH'97*. 1997. ACM. pp. 19-26.
Keywords: *3D interfaces, virtual reality, interaction techniques, manipulation, navigation, proprioception, 3D widgets, guidelines*
Annotations: 1. Thorough analysis of the effects of proprioception on user interface techniques. 2. Discuss design of a interaction techniques for manipulation, navigation and other tasks. Introduces several new techniques and discuss their design implications.
102. Osborn, J., Agogino, A., An Interface for Interactive Spatial Reasoning and Visualization. *Proceedings of Human Factors in Computing Systems Conference*. 1992. HFS. pp. 75-82.
Keywords: *Spatial reasoning, cutting planes, desktop 3D, visualization*
Annotations: This paper describes a mouse-based user interface for spatial reasoning and visualization. The interface includes the ability to orient an object and select arbitrary cutting planes; this portion of the interface is discussed in considerable detail. The basic interaction metaphor is that of manipulating the object in a "pool of water," the surface of which forms the cutting plane. The user rotates the model into the desired orientation, and then adjusts the depth of the "pool" to select the depth of the cut.
103. Ostby, E., Describing Free-Form 3D Surfaces for Animation. *Proceedings of Workshop on Interactive 3D Graphics*. 1986. ACM. pp. 251-258.
Keywords: *Tracking, freeform modeling, 3D user interfaces, surface deformation.*
Annotations: The author investigates several uses of the Polhemus tracker for specifying the surfaces of 3D objects. Uses included: * Probe: Sample probe at user signal; sample at many points in space to define an object. It was hard to locate a 3D point on a two-dimensional display. The author found that using the device in combination with a real object helped solve this problem. 1.) Pencil: Draw lines in space, or trace a grid over the surface of an actual object. Use least-squares to fit a patch 2.) Camera for viewing: Works well and feels natural 3.) Tool for deforming the surface of existing objects: Use relative motion to deform. Relative motion is easier to control. Problems with the Polhemus tracker included: * Lack of a tip

switch -- need equivalent of mouse click * Drawing free-hand in open space is hard: no friction to facilitate control * Locating points in space with only 2D display

104. Pausch, R., Burnette, T., Brockway, D., Weiblen, M., Navigation and Locomotion in Virtual Worlds via Flight into Hand-Held Miniatures. *Proceedings of SIGGRAPH'95, Technical Sketches*. 1995. ACM. pp. 399-400.
Keywords: *multi-scale travel, viewpoint manipulation, interaction technique, virtual reality*
Annotations: This short paper describes a technique for moving one's viewpoint by moving a small icon representing yourself, and then flying into the scale model to take that new position.
105. Pausch, R., Burnette, T., Brockway, D., Weiblen, M., Navigation and Locomotion in Virtual Worlds via Flight into Hand-Held Miniatures. *Proceedings of SIGGRAPH'95*. 1995. ACM. pp. 399-400.
Keywords: *3D interfaces, navigation, locomotion, WIM, VR*
Annotations: This paper describes the use of a World-in-Miniature (WIM) as a navigation and locomotion device in immersive virtual environments. When the user moves an iconic representation of himself in the WIM he moves (flies) in the virtual environment. See also [Stoackley, et al. 1995]
106. Pausch, R., Crea, T., Conway, M., A Literature Survey for Virtual Environments: Military Flight Simulator Visual Systems and Simulator Sickness. 1995. 1(3).
Keywords: *motion sickness, VR, flight simulators, military, survey*
Annotations: Gives a quick overview of simulator research along with lots of references from military research which are very difficult to find in the academic literature. The references are annotated.
107. Pausch, R., Snoddy, J., Taylor, R., Watson, S., Haseltine, E., Disney's Aladdin: first steps toward storytelling in virtual reality. *Proceedings of SIGGRAPH'96*. 1996. ACM.
Keywords: *virtual reality, story telling, evaluation, navigation, design, art, presence, immersion, VR entertainment*
Annotations: The paper describes experience of developing and results of evaluation a virtual reality entertainment attraction "Aladdin" installed in Disney park. Results of observation and interview with approximately 45000 visitors over 14 months are reported. Many important interface issues related to design of VR entertaining systems are reported here.
108. Pierce, J., Conway, M., Van Dantzich, M., Robertson, G., Toolspaces and Glances. *Proceedings of Symposium on Interactive 3D Graphics*. 1999. pp. 163-168.
Keywords: *Gestural interaction, viewpoint control, tools*
Annotations: This paper presents "two go greets that go great together": glances, a lightweight approach to providing different viewpoints, and toolspaces, a way of using the space around the user for different purposes. Together these techniques allow designers to, for example, create a 3D storage space that can be accessed with a simple gesture for users of desktop 3D worlds.
109. Pierce, J., Forsberg, A., Conway, M., Hong, S., Zeleznik, R., et al., Image plane interaction techniques in 3D immersive environments. *Proceedings of Symposium on Interactive 3D Graphics*. 1997. ACM. pp. 39-43.
Keywords: *3D, manipulation, selection, interaction technique, VR*
Annotations: An interesting approach to the problem of remote object manipulation in immersive VEs.
110. Pierce, J., Stearns, B., Pausch, R., Two Handed Manipulation of Voodoo Dolls in Virtual Environments. *Proceedings of Symposium on Interactive 3D Graphics*. 1999. pp. 141-145.
Keywords: *Image plane techniques, selection, voodoo dolls, two handed interaction*
Annotations: The Voodoo Dolls technique leverages some of Pierce's earlier work on image plane techniques to provide a method of fluidly manipulating and positioning objects in a scene. The technique draws on how users work with their hands, where the non-dominant hand defines the frame of reference that the dominant hand works in.
111. Poston, T., Serra, L., Dextrous Virtual Work. *Communications of the ACM*, 1996. 39(5): pp. 37-45.
Keywords: *application, medicine, 3D interaction, two-handed manipulation, visualization*
Annotations: "A system for visualizing and manipulating medial images is detailed, with emphasis on interaction techniques." Uses a mirrored set up (opaque mirror, not half-silvered) with stereoscopic display. The mirror is up relatively near your face, leaving a large work volume for the hands behind the mirror. Not a major point of the paper, but the system employs two-handed interaction: rotation of a brain image with the left hand and fine manipulation with the right hand, using a physical tool handle that has multiple virtual effectors.
112. Poupyrev, I., Billingham, M., Weghorst, S., Ichikawa, T., Go-Go Interaction Technique: Non-Linear Mapping for Direct Manipulation in VR. *Proceedings of UIST'96*. 1996. ACM. pp. 79-80.
Keywords: *3D interaction, 3D user interface, direct manipulation, interaction techniques, virtual reality, non-linear C-D gain*
Annotations: Interaction technique for manipulation in virtual environment. The technique uses non-linear control display gain to extend the user reach and manipulate both objects within the user reach as well as those at a distance.
113. Poupyrev, I., Tomokazu, N., Weghorst, S., Virtual Notepad: handwriting in immersive VR. *Proceedings of VRAIS'98*. 1998. IEEE. pp. 126-132.
Keywords: *virtual environment, interaction technique, handwriting, text input, annotation, pen input, 2D input in 3D, multimodal interaction*
Annotations: The Virtual Notepad is a set of interface tools that uses handwriting to allow the user to write, draw and annotate documents while immersed in virtual environments. Using handwriting recognition with Virtual Notepad allows for text input from within virtual environments.
114. Poupyrev, I., Weghorst, S., Billingham, M., Ichikawa, T., A framework and testbed for studying manipulation techniques for immersive VR. *Proceedings of VRST'97*. 1997. ACM. pp. 21-28.
Keywords: *VR, manipulation, 3D user interfaces, testbed, experimental evaluation, user study, taxonomy, design, task analysis*

- Annotations:** Testbed for experimental evaluation of 3D interaction techniques in virtual environments: Virtual Reality Manipulation Assessment Testbed (VRMAT). The VRMAT provides a detailed break-down of the 3D manipulation task, discusses main variables affecting user manipulation performance and attempts to introduce user-centered metrics for describing spatial characteristics of the virtual environments. The main idea behind introducing user-centered metrics, i.e. metrics based on the user physical parameters, such as arm length, rather than on the physical parameters of the real world, such as in case of meters, is to be able to describe the spatial characteristics of the environment independently from the system implementation and from the point of view of the user
115. Poupirev, I., Weghorst, S., Billingham, M., Ichikawa, T., Egocentric object manipulation in virtual environments: empirical evaluation of interaction techniques. *Computer Graphics Forum, EUROGRAPHICS'98 issue*, 1998. 17(3): pp. 41-52.
Keywords: *interaction techniques, taxonomy, classification, user studies, experiments, manipulation, pointing and grabbing, metaphor*
Annotations: Describes the first formal experiments to evaluate interaction techniques for object selection and manipulation in immersive VEs. The experiments were conducted within the taxonomy of interaction techniques which is also presented in the paper.
116. Poupirev, I., Weghorst, S., Fels, S., Non-isomorphic 3D rotational interaction techniques. *Proceedings of CHI2000*. 2000. ACM.
Keywords: *3D interaction, VR, object manipulation, 3D input devices, device form factor, interaction techniques, object rotations, user studies*
Annotations: This paper demonstrates how to design 3D rotational interaction techniques that amplify rotations of the input device in manipulation task. It also demonstrates their properties and shows how these techniques can be used to build effective spatial 3D user interfaces. First, a mathematical framework allowing to design 3D rotational mappings and techniques, is designed. Then, authors investigate their usability properties. Finally, the experimental results are reported. The results suggest that non-isomorphic rotational mappings can be effective in building 3D manipulation dialogues, in reported experiments they allowed subjects to accomplish experimental tasks 13% faster without a statistically detectable loss in accuracy.
117. Raab, F., Blood, E., Steiner, T., Jones, H., Magnetic Position and Orientation Tracking System. *IEEE Transaction on Aerospace and Electronic Systems*, 1979. AES-15(5): pp. 709-717.
Keywords: *3D input device, magnetic tracker, Polhemus, position and orientation tracking*
Annotations: Describes the theoretical underpinnings of the Polhemus tracker. Also talks about application considerations, source/sensor imperfections, and the problems caused by nearby metallic structure. Rule of thumb for metal: "An object whose distance from the source is at least twice the distance separating the source and sensor produces a scattered field whose magnitude is 1 percent or less of the magnitude of the desired field."
118. Regenbrecht, H., Schubert, T., Friedmann, F., Measuring the Sense of Presence and its Relations to Fear of Heights in Virtual Environments. *International Journal of Human-Computer Interaction*, 1998. 10(3): pp. 233-250.
Keywords: *presence, measurement, fear of heights, VR*
Annotations: Good article dealing with measuring presence, the thereby involved factors which cause presence, and the coupling to fear of heights tests.
119. Robbins, D., Practical 3D user interface design. SIGGRAPH'96 Course Notes: *Technical Report 31*. 1996.
Keywords: *3D user interfaces, interface design, desktop 3D interfaces*
Annotations: These notes for tutorial presented at the SIGGRAPH'96 take the reader through the process of designing 3D interface for a simple application: Theater Lightning Design. The application and 3D interface are being developed for the conventional desktop 3D environment. Tutorial discusses many ideas, rules and tricks in designing effective 3D interaction.
120. Robertson, G., Card, S., Mackinlay, J., The Cognitive Coprocessor Architecture for Interactive User Interfaces. *Proceedings of UIST'89*. 1989. ACM. pp. 10-18.
Keywords: *3D interfaces, software tool, software architecture, agent, animation*
Annotations: Describes a software architecture which is appropriate for the real-time demands of 3D interactive applications, including animation. There are two problems: * Multiple Agent Problem: UI must match "time constants" of human and computer. The architecture must "manage the interactions of multiple asynchronous agents that can interrupt and redirect each other's work." * Animation Problem: Interactive animation can shift the user's task from cognitive to perceptual, which fees cognitive ability. Animation of motion allows a continuity of perception; discontinuous motion requires reassimilation of the new display. The paper advocates a three agent model: User, user discourse machine, and task machine. The cognitive coprocessor is a UI architecture which supports this model, plus "intelligent" agents and smooth animation. The animation loop (on the user discourse machine) is the basic control mechanism. It maintains a task queue (pending computations from agents), a display queue (pending instructions from against for how the screen should be painted on the next animation loop cycle), and a governor (keeps track of time & allows for adjustments to animations to keep them smooth). The paper goes on to describe a "3D Rooms" example based on this architecture.
121. Ruddle, R., Payne, S., Jones, D., Navigating large-scale "Desk-Top" virtual buildings: Effects of orientation aids and familiarity. *Presence: Teleoperators and Virtual Environments*, 1998. 7(2): pp. 179-192.
Keywords: *Wayfinding, spatial orientation, cues, aids, desktop VR*
Annotations: Excellent article dealing with how participants deal with spatial knowledge when navigating in virtual worlds using a desktop VR system. Article describes 2 experiments which deal with direction and distance estimations. Authors describe influences of aids (like a compass) on the accuracy of spatial knowledge.
122. Rygol, M., Dorbie, A., Worden, A., Ghee, S., Grimsdale, C., et al., Tools and metaphors for user interaction in virtual environments. In *Virtual Reality Applications*, R.A. Earnshaw, J.A. Vince, and H. Jones, Editors. 1995, Academic Press. pp. 149-161.

- Keywords:** 3D interaction, interface toolkit, 3D user interface, 3D widgets, direct manipulation, VR tools, metaphor
- Annotations:** Describes 3D user interface toolkit that has been implemented in dVS, which is a commercial software system for building virtual environments (by Division Ltd). The toolkit is based on the collection of 3D widgets. The paper describes goals of development of the toolkit, logical organization of widgets and their properties and finally describes some of the widgets implemented. The paper is interesting at least because this work has become a real commercial product.
123. Sachs, E., Roberts, A., Stoops, D., 3-Draw: A Tool for Designing 3D Shapes. *IEEE Computer Graphics and Applications*, 1991. pp. 18-26.
Keywords: two-handed interfaces, 3D user interfaces, sketching, 3D modeling
Annotations: This paper describes a system for "sketching" in three dimensions using a pair of Polhemus 3Space trackers. A palette is held in one hand. A stylus held in the other hand is moved relative to it, allowing the user to sketch curves in 3D. Thus the interaction is based on two-handed manipulation of tools on "props."
124. Schmalsteig, D., Encarnacao, L., Szalczvari, Z., Using Transparent Props For Interaction with The Virtual Table. *Proceedings of Symposium on Interactive 3D Graphics*. 1999. ACM. pp. 147-154.
Keywords: transparent props, through the plane tool, Barco Table, VR, 3D interfaces, interaction techniques
Annotations: This paper presents a number of interaction techniques using a tracked, transparent pad. This tool is a mechanism for entering 2D input in a 3D virtual environment.
125. Schmandt, C., Spatial Input/Display Correspondence in a Stereoscopic Computer Graphic Work Station. *Proceedings of SIGGRAPH'83*. 1983. ACM. pp. 253-262.
Keywords: VR, 3D interfaces, desktop, displays, stereoscopy, depth cues, interaction, manipulation, wand
Annotations: This paper describes a work station designed to allow interaction with spatial correspondence between the input (Polhemus) and output (stereoscopic display) devices. The workspace consists of a monitor mounted at a 45 degree angle and a half-silvered mirror, beneath which the user holds the "wand." This set-up mixes the computer graphics and the user's hand into a single image. Pure binocular convergence was found to lack sufficient depth cues. A combination of convergence, obscurations, luminance, and size give a strong 3D sense, but no factor alone was adequate. Schmandt reports that a significant problem was lack of depth judgement. Occlusion cues were misleading, as the user could always see their hand through the semi-transparent graphics.
126. Sebrechts, M., Vasilakis, J., Miller, M., Cugini, J., Lasowski, S., Visualization of Search Results: A Comparative Evaluation of Text, 2D, and 3D Interfaces. *Proceedings of SIGIR'99*. 1999. pp. 3-10.
Keywords: 3D interface, visualization, user studies, experimental evaluation
Annotations: The paper describes a controlled experiment to measure the effectiveness of the user interface for a system that categorizes and presents the set of documents resulting from an automatic search. 3-D, 2-D and textual variants of the interface were developed and compared.
127. Serra, L., Poston, T., Hern, N., Choon, C., Waterworth, J., Interaction techniques for a virtual workspace. *Proceedings of VRST'95*. 1995. ACM.
Keywords: 3D interaction techniques, system control, widgets, responsive workbench
Annotations: Good overview of a large amount of 3D interaction techniques implemented on a responsive workbench. The authors also report on several implementations of techniques in medical applications.
128. Shaw, C., Green, M., Two-Handed Polygonal Surface Design. *Proceedings of UIST'94*. 1994. ACM. pp. 205-212.
Keywords: desktop VR, 3D user interface, manipulation, two-handed input
Annotations: Describes a system which uses two hand-held trackers (augmented with 3 buttons each) to perform CAD tasks. The dominant hand performs picking and manipulation, the non-dominant hand context setting.
129. Shaw, C., Liang, J., Green, M., Sun, Y., The Decoupled Simulation Model for Virtual Reality Systems. *Proceedings of CHI'92*. 1992. ACM.
Keywords: software tools, virtual reality, software architectures, development toolkits
Annotations: Describes the MR toolkit
130. Shepherd, B., Rationale and strategy for VR standards. *Proceedings of VRAIS'93*. 1993. IEEE. pp. 41-46.
Keywords: VR, standards, survey, directions, 3D interaction, VR tools.
Annotations: This paper discusses the need and rationale for standards in VR field: why standards are needed, what should be standardized, and what are the current (i.e. in 1993) efforts in developing standards. The author suggests that standards for VR will stabilize VR market, because it will help the software developers and hardware producers in easier integration of different products. It will also help customers since it will allow the purchase of technology without or at least with less worries technology will become obsolete, incompatible and unusable as the VR technology is developed. The areas that would benefit from standardization, according to the author, are user interfaces and software interfaces.
131. Shoemake, K., ARCBALL: a user interface for specifying three-dimensional orientation using a mouse. *Proceedings of Graphics Interface'92*. 1992. pp. 151-156.
Keywords: desktop 3D user interface, 3D rotations, interaction technique, mouse
Annotations: Describes a 2D interface for 3D orientation. The key is that mouse motion is consistently interpreted as a half-arc length rotation on an imaginary sphere, resulting in an interface free from hysteresis. A circle is drawn around the object being rotated; rotation about the axis perpendicular to the screen is handled by moving the mouse in the region outside of this circle. The paper also demonstrates how to add constrained rotations to the technique.
132. Sims, D., Osmose: Is VR Supposed to be this Relaxing? *Computer Graphics and Applications*, 1996. 16(6): pp. 4-5.
Keywords: VR, art, navigation
Annotations: Osmose was one of the first and one of the most compelling art-oriented VR environments. According to one of the authors: "Osmose is a space for exploring

- the perceptual interplay between self and world, i.e. a place for facilitating awareness of one's own self as embodied consciousness in enveloping space. " From the interface point of view, the Osmose introduced an interesting navigating technique that used balance and breathing for "floating" in the VE. The user was wearing a device on the chest which allowed to detect the breathing patterns. By breathing in, the user was able to float upward, by breathing out, to fall, and by altering the body's center of balance, to change direction. The technique was borrowed from the scuba diving practice of buoyancy control.
133. Slater, M., Davidson, A., Liberation from Flatland: 3D Interaction Based on the Desktop Bat. *Proceedings of Eurographics '91*. 1991. Elsevier Science Publishers B.V. (North Holland). pp. 209-221.
Keywords: 3D interaction, interaction devices, desktop virtual environments
Annotations: An early example of a desktop control device designed for 3D tasks. A 5dof device consisting of a dome on top of a mouse base. Discussion of utility of such a device for 3D control tasks.
134. Slater, M., Steed, A., 3D Interaction with the Desktop Bat. *Computer Graphics Forum*, 1995. 14(2): pp. 97-104.
Keywords: 3D interaction, interaction devices, virtual environments, bat
Annotations: The paper presents interaction methods for locomotion, selection and manipulation that can be used with a 5dof desktop control device, the Desktop Bat. These methods are evaluated over a series of characteristic VE tasks.
135. Slater, M., Usoh, M., Body Centred Interaction in Immersive Virtual Environments. In *Virtual Reality and Artificial Life*, M. Magnenat-Thalmann and D. Thalmann, Editors. 1994, John Wiley.
Keywords: body interaction, natural gestures
Annotations: The authors propose gestures made by the whole body as a consistent metaphor for interaction within virtual environments.
136. Slater, M., Usoh, M., Steed, A., Taking Steps: The influence of a walking metaphor on presence in virtual reality. *ACM Transactions on Computer-Human Interaction*, 1995. 2(3): pp. 201-219.
Keywords: presence, navigation, wayfinding, physical movement, VR
Annotations: Authors describe how physically walking influences the feeling of presence in a virtual environment. The article also deals with the usage of a virtual body in a virtual environment, and its effects on presence.
137. Slater, M., Usoh, M., Steed, A., Taking Steps: The Influence of a Walking Technique on Presence in Virtual Reality. *Transactions on Computer-Human Interaction*, 1995. 2(3): pp. 201-219.
Keywords: locomotion, travel, natural interaction, virtual environment, presence, immersion
Annotations: A study on whether a more natural movement technique increases presence
138. Smith, S., Duek, D., MAssink, M., The Hybrid World of Virtual Environments. *Computer Graphics Forum*, 1999. 18(3): pp. 297-308.
Keywords: interaction techniques, software engineering, hybrid systems, prototyping
- Annotations:** The authors suggest that virtual environments can be described within a hybrid system model. They discuss how hybrid system modeling techniques can be applied to the description of virtual interaction techniques.
139. Snibbe, S., Herndon, K., Robbins, D., Using deformations to explore 3D widget design. *Proceedings of SIGGRAPH'92*. 1992. ACM. pp. 351-352.
Keywords: 3D widgets, 3D interaction, 3D user interfaces, interactive deformations.
Annotations: 1. Early article on 3D widget design issues, showing a set of new 3D widgets to control deformations called racks. 2. One of the widget papers from Brown CG group. See also Conner, et al. 1992 and Zeleznik, et al. 1994
140. Song, D., Norman, M., Nonlinear interactive motion control techniques for virtual space navigation. *Proceedings of VRAIS'93*. 1993. IEEE. pp. 111-117.
Keywords: virtual reality, navigation, 3D user interfaces, interaction techniques, viewpoint control
Annotations: The paper presents an interaction technique for navigation in virtual environments, using function that non-linearly maps displacement of the head into the traveling speed.
141. Stanney, K., Realizing the full potential of virtual reality: human factors issues that could stand in the way. *Proceedings of VRAIS'95*. 1995. IEEE Computer Society. pp. 28-34.
Keywords: 3D interaction, conceptual models, experimental studies, human factors, user interface, virtual reality, VR, simulator sickness, survey
Annotations: A survey paper describing various VR systems and techniques with respect to their adherence to human factors guidelines.
142. Stassen, H., Smets, G., Telemanipulation and Telepresence. *Proceedings of 6th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design, and Evaluation of Man-machine System*. 1995. pp. 13-23.
Keywords: 3D manipulation, telepresence, survey, theory
Annotations: A survey of 3D manipulation and perception work from the teleoperation point of view. Touches on: 3D perception (Softenon children: without manipulation, children don't develop 3 perception), theories of perception, television (adapting to teleoperation tasks), telemanipulation: handedness, field of view & depth, time delay, implementation. Interesting work in handedness has been done in the rehabilitation field, especially in design of arm prostheses.
143. Stevens, M., Zeleznik, R., Hughes, J., An architecture for an extensible 3D interface toolkit. *Proceedings of UIST'94*. 1994. ACM. pp. 59-67.
Keywords: user interface toolkit, software toolkit, visual programming, 3D widget, 3D user interfaces, 3D interaction, interaction techniques, direct manipulation
Annotations: Describes the architecture and implementation details of the 3D user interface toolkit developed at Brown (see Zeleznik, et al., 1993).
144. Stoakley, R., Conway, M., Pausch, R., Virtual reality on a WIM: interactive worlds in miniature. *Proceedings of CHI'95*. 1995. pp. 265-272.

- Keywords:** 3D interaction, 3D user interface, 3D widgets, direct manipulation, interaction techniques, navigating, user interface, viewpoint control, virtual reality, VR.
- Annotations:** 1. The original WIM (World In Miniature) paper describing the technique for manipulating world objects by moving small iconic versions of those objects in a scale model of the world 2. Describes the Worlds in Miniature interface metaphor. Augments an immersive head tracked display with a hand held miniature copy of the virtual environment; there is a 1:1 relationship between life-size objects in the virtual world and miniature objects on the hand-held miniature world.
145. Strauss, P., Carey, R., An object-oriented 3D graphics toolkit. *Proceedings of SIGGRAPH'92*. 1992. ACM. pp. 341-347.
Keywords: 3D interaction, Open Inventor, 3D interface toolkit, 3D user interface, 3D widgets, direct manipulation, interaction techniques, 3D computer graphics, graphics toolkit, scene graph
Annotations: This is an original paper that reported software toolkit that become SGI Open Inventor tool. The toolkit pioneered using scene graph of scene representation as well as incorporation of interaction techniques in the toolkit: the toolkit featured handle boxes, techniques for interactive 3D object rotation and other techniques. The techniques, however, are designed for the desktop 3D interaction using mouse.
146. Stuart, R., *The design of virtual environments*. 1996: McGraw Hill. pp. 274.
Keywords: virtual reality, design, 3D input and output devices, interaction techniques, human factors.
Annotations: This book survey a large range of issues relating to design and implementation of virtual reality applications, including devices, interaction, human factors, social factors, software tools and so on. The book is somewhat outdated when it comes to interaction techniques, but still it is a good survey.
147. Sutherland, I., The Ultimate Display. *Proceedings of IFIP Congress*. 1965. pp. 505-508.
Keywords: VR, head-mounted display, HMD
Annotations: Another seminal VR paper
148. Sutherland, I., A Head-mounted Three Dimensional Display. *Proceedings of Fall Joint Computer Conference*. 1968. pp. 757-764.
Keywords: VR, HMD
Annotations: The seminal VR paper -- discusses technical details of Sutherland's original see-through display system
149. Szalavari, Z., Gervautz, M., The Personal Interaction Panel - a Two-Handed Interface for Augmented Reality. *Computer Graphics Forum*, 1997. 16(3): pp. 335-346.
Keywords: Two-handed interaction, augmented reality, pen and pad interfaces
Annotations: A tracked pen and pad input device that supports interaction with different types of virtual widgets. Some example applications are given.
150. Takemura, H., Tomono, A., An Evaluation of 3-D Object Pointing Using a Field Sequential Stereoscopic Display. *Proceedings of Graphics Interface'88*. 1988. pp. 112-118.
Keywords: experimental studies, pointing, stereoscopy, depth effect
- Annotations:** Describes user experiments (six subjects) to measure performance in 3-D object pointing with stereoscopy.
151. Tarlton, M., Tarlton, P., A Framework for Dynamic Visual Applications. *Proceedings of Symposium on Interactive 3D Graphics*. 1992. ACM. pp. 161-164.
Keywords: Software toolkit, architecture
Annotations: Describes the Mirage system, a precursor to Inventor
152. Thorndyke, P., Hayes-Roth, B., Differences in spatial knowledge obtained from maps and navigation. *Cognitive Psychology*, 1982. 14: pp. 560-589.
Keywords: Spatial knowledge, navigation, cues
Annotations: This article, although it does not directly deal with VR and 3D interfaces, is one of the major references for most of the theories concerning spatial knowledge, wayfinding, spatial awareness within virtual environments. The authors developed a strong framework and especially their map-test is noteworthy.
153. Turner, R., Gobbetti, E., Soboroff, I., Head-tracked Stereo-Viewing with Two-Handed Interaction for Animated Character Construction. *Computer Graphics Forum*, 1996. 15(3): pp. 197-206.
Keywords: Two-handed input, virtual tools, character modeling
Annotations: A combination of a Spaceball and a 3D mouse are used to manipulate virtual tools for the construction of animated characters.
154. Usoh, M., Arthur, K., Whitton, M.C., Bastos, R., Steed, A., Slater, M., Brooks, F.P. Jr, Walking > Walking-in-Place > Flying in Virtual Environments. *Proceedings of SIGGRAPH '99*. 1999. ACM. pp. 359-364.
Keywords: Presence, locomotion, virtual reality, virtual walking, human factor, neural networks, visual cliff
Annotations: Comparison of three locomotion techniques for virtual reality: point and fly, virtual walking (virtual treadmill) and real walking using a wide area ceiling tracker. Virtual walking and real walking are significantly different from point and fly.
155. Verlinden, J., Bolter, J., der-Mast van, C., Virtual annotation: verbal communication in virtual reality. *Proceedings of European Simulation Symposium*. 1993. SCS Ghent, Belgium. pp. 305-310.
Keywords: virtual annotation, verbal communication, virtual reality, simulations, visualizers, voice annotation, virtual environments
Annotations: 1. The paper describes a system that offers a method to embed verbal communication in virtual environments by means of voice annotation. The prototype demonstrates that the addition of verbal communication opens up a range of new uses for virtual environments and it enables reading, writing and communicating. 2. The paper describes an idea and a prototype. More complete implementation of this work is described later in Harmon, et al. 1996
156. Viega, J., Conway, M., Williams, G., Pausch, R., 3D Magic Lenses. *Proceedings of UIST'96*. 1996. ACM. pp. 51-58.
Keywords: Magic Lenses, transparent user interfaces, 3D interaction, virtual reality, VR, clipping.
Annotations: Good article on porting Magic Lenses idea to a 3D environment. Two types of Magic Lenses was in-

- roduced: flat lenses in 3D environment, and volumetric lenses in 3D environment.
157. Waller, D., Hunt, E., D., K., Measuring spatial knowledge in a virtual environment: Distances and angles. *Proceedings of 39th Annual Meeting of the Psychonomics Society*. 1998. pp. 129-143.
Keywords: *spatial knowledge, wayfinding, distance and angle estimation*
Annotations: Article on the differences of distance and angle estimations within real and virtual environments [see also Henry, 1993]
158. Waller, D., Hunt, E., Knapp, D., The transfer of spatial knowledge in virtual environment training. *Presence: Teleoperators and Virtual Environments*, 1998. 7(2): pp. 129-143.
Keywords: *Wayfinding, spatial knowledge, knowledge transfer, VR*
Annotations: Noteworthy article on the transfer of spatial knowledge between virtual environments and the real world. Focuses especially on distance angle biases.
159. Ware, C., Using hand position for virtual object placement. *Visual Computer*, 1990. 5(6): pp. 245-253.
Keywords: *3D interface, interaction techniques, manipulation, VR, multiple DOF*
Annotations: This paper describes two experiments which investigate the use of six degree of freedom digitizers (Polhemus) to manipulate 3D virtual environments. Specifically, the experiments test the speed and accuracy of placing an object in space with the correct orientation. Motions always had a total magnitude of 9.5 cm. Four subjects participated in the study. In the first experiment subjects were told to position the object (both position and orientation) as accurately as possible. Four conditions were tested: z translation enabled, z disabled, stereo, no stereo. Enabling z translations slowed accurate placement: 25% with stereo, 53% without stereo. Overall, the placement times with stereopsis were 39% faster. In the second experiment, subjects were told to make the placement as quickly as possible. Times to position, orient, or (simultaneously) position and orient were tested. Ware found that subjects were able to make effective use of all six degrees of freedom (that is, time for simultaneous positioning & orientation was less than the time for separate positioning and orientation). Disabling the z translations hindered rapid placement. Stereopsis still helped. Subjects did not report fatigue with the Polhemus. Ware states this is because it was used as a relative positioning device.
160. Ware, C., Jessome, D., Using the bat: a six-dimensional mouse for object placement. *IEEE Computer Graphics&Applications*, 1988. 8(6): pp. 65-70.
Keywords: *computer graphics. mice. computer graphic equipment. six- dimensional mouse. object placement. 6-D placement. bat. placement operations. hierarchically constructed scene. visualization. manipulation. software environment.*
Annotations: 1. Reports on experiments with a Polhemus tracker. Summary of some interesting points: It is essentially impossible to achieve precise positioning using a 1:1 control ratio when the arm/hand is unsupported. Rotations of the Polhemus produce inadvertent translations. Interaction techniques which require the user to precisely control both sets of parameters simultaneously are "generally confusing." Uses "ratcheting" for large translations or rotations: a button on the bat acts as a clutch allowing or disallowing movement. 2. The pioneering research on usability of 3D input devices.
161. Ware, C., Osborne, S., Exploration and Virtual Camera Control in Virtual Three Dimensional Environments. *Proceedings of Symposium on Interactive 3D Computer Graphics*. 1990. ACM. pp. 175-183.
Keywords: *viewpoint movement, travel*
Annotations: 1. A paper surveying the early state of the art in travel techniques for 3D environments. 2. Discusses basic interaction paradigms for 3D data. Metaphors: eyeball in hand, Scene in hand, Flying vehicle control. Studies where they are appropriate.
162. Ware, C., Rose, J., Rotating virtual objects with real handles. *Transaction on Computer-Human Interaction*, 1999. 6(2): pp. 162-180.
Keywords: *3D user interface, manipulation, rotation, virtual reality, feedback, user studies, experiments*
Annotations: Interesting studies which compared user performance in object rotation task in real and virtual worlds. A number of conditions have been varied, such as two-handed versus one-handed, matching shape of virtual and physical objects versus non-matching shape and so on. They founded 1) the rotation in the same space, i.e. when physical hand and virtual hand "overlap", is more effective 2) it was not important to match shape of virtual and physical objects 3) there was no advantage of two-handed manipulation over one-handed manipulation 4) the virtual rotation can be performed almost as quickly as real if conditions are right. Data they report can be very useful as a comparison with other studies.
163. Ware, C., Slipp, L., Exploring virtual environments using velocity control: A comparison of three interfaces. *Proceedings of 35th Annual Meeting of Human Factors Society*. 1991. HFS. pp. 300--304.
Keywords: *VR, 3D interfaces, interaction techniques, navigation*
Annotations: Compares Polhemus, Spaceball, and mouse-based interfaces. Spaceball yielded worst performance. Some users complained of fatigue after prolonged use of the Polhemus, but it still yielded the best results.
164. Watsen, K., Darken, R., Capps, M., A Handheld Computer as an Interaction Device to a Virtual Environment. *Proceedings of Third Immersive Projection Technology Workshop*. 1999.
Keywords: *palm pilot, virtual environment interaction, mobile computing*
Annotations: This paper describes how a PDA can be used to issue commands in a virtual environment such as a Cave.
165. Weimer, D., Ganapathy, S., A synthetic visual environment with hand gesturing and voice input. *Proceedings of CHI'89*. 1989. ACM. pp. 235-240.
Keywords: *Gestural input, voice input, feedback*
Annotations: Talks about glove + voice input. Their focus is on development of synthetic environment interaction techniques, as a vehicle for experimenting with more natural 3D interfaces. A table top is used as a workspace, giving a place to rest the hands, and also providing a sort of "natural" tactile feedback when "buttons" are pressed on a menu in the synthetic space. A standard monitor is used for display. Speech input was added to the interface

for three reasons: (1) people tend to use gestures to augment speech, (2) spoken vocabulary has a more standard interpretation than gestures, (3) hand gesturing and speech complement one another. Voice is used for navigating through commands, while hand gestures provide "shape" information. "There was a dramatic improvement in the interface after speech recognition was added." A thumb gesture is used as a clutching mechanism to avoid uncomfortable hand positions. The driving application is a 3D modeling system for free-form surfaces.

166. Witmer, B., Bailey, J., Knerr, B., Parsons, K., Virtual spaces and real world places: Transfer of route knowledge. *International Journal of Human-Computer Studies*, 1996. 45: pp. 413-428.
Keywords: *spatial knowledge, route knowledge, knowledge transfer*
Annotations: Great article on knowledge transfer between virtual and real environments in training, especially route knowledge.
167. Wloka, M., Interacting with Virtual Reality. In *Virtual Prototyping - Virtual Environments and the Product Development Process*, J. Rix, S. Haas, and J. Teixeira, Editors. 1995, Chapman & Hall.
Keywords: *Interaction, user interface, performance*
Annotations: The paper defines the three features that characterize virtual reality applications: immersion, rich interaction and presence. Some of the issues to achieve them are discussed, in particular multiple inputs and outputs, multiple participants, dynamic virtual worlds, user interface paradigms and performance.
168. Wloka, M., Greenfield, E., The virtual tricorder: a uniform interface for virtual reality. *Proceedings of UIST'95*. 1995. ACM. pp. 39-40.
Keywords: *interaction metaphors, commands in VEs, selection, manipulation, navigation, interaction techniques*
Annotations: 1. An attempt to create a single interaction framework for VEs using a virtual instrument. 2. Technique implements the idea of Virtual Tricorder: a universal interaction device for virtual reality which was first suggested by Henry Sowitzal
169. Youngblut, C., Johnson, R., Nash, S., Wienclaw, R., Will, C., Review of Virtual Environment Interface Technology. Institute for Defense Analysis: *Technical Report IDA Paper P-3186, Log: H96-001239*. 1996.
Keywords: *input devices, display devices, virtual reality hardware*
Annotations: Although a little dated, this paper presents a comprehensive list of input and output devices for 3D user interfaces.
170. Zeleznik, R.C., Herndon, K.P., Hughes, J.F., SKETCH: an interface for sketching 3D scenes. *Proceedings of SIGGRAPH'96*. 1996. ACM. pp. 163-70.
Keywords: *SKETCH, 3D user interface, 3D scene sketching, non photorealistic rendering, gestural interface, pen and tablet input*
Annotations: 1. The SKETCH application described in the paper allows for rapidly conceptualizing and editing approximate 3D scenes. To achieve this, SKETCH uses simple non photorealistic rendering and a purely gestural interface based on simplified line drawings of primitives that allows all operations to be specified within the 3D world. 2. The classical work of using 2D gestures for 3D modeling. The basic idea is to allow the user to sketch 3D scenes using a small set of 2D gestures, that he or she simply draw on the tablet.
171. Zeleznik, R.C., Herndon, K.P., Robbins, D.C., Huang, N., Meyer, T., *et al.*, An interactive 3D toolkit for constructing 3D widgets. *Proceedings of SIGGRAPH'93*. 1993. ACM. New York, NY, USA. pp. 81-4.
Keywords: *interactive 3D toolkit, 3D widgets, deformation racks, interactive shadows, parameterized models, 3D geometries, interactive toolkit, visual programming, interactive models, interactive behavior, three dimensional widget.*
Annotations: 1. This is the first attempt to create a software toolkit for designing 3D user interfaces. The resulted 3D interfaces are based on heavy use of 3D widgets, which were also introduced by this CG group at Brown University.
172. Zhai, S., Buxton, W., Milgram, P., The "Silk cursor": investigating transparency for 3D target acquisition. *Proceedings of CHI'94*. 1994. ACM. pp. 459-464.
Keywords: *3D selection, cursor, 3D interface, desktop VR, volumetric interface*
Annotations: 1. From paper abstract: This study investigates dynamic 3D target acquisition. The focus is on the relative effect of specific perceptual cues. A novel technique is introduced and we report on an experiment that evaluates its effectiveness. There are two aspects to the new technique. First, in contrast to normal practice, the tracking symbol is a volume rather than a point. Second, the surface of this volume is semitransparent, thereby affording occlusion cues during target acquisition. The experiment shows that the volume/occlusion cues were effective in both monocular and stereoscopic conditions. For some tasks where stereoscopic presentation is unavailable or infeasible, the new technique offers an effective alternative. 2. It was the first paper that investigated volumetric selection cursors as well as semitransparent cursors.
173. Zhai, S., Milgram, P., Buxton, W., The influence of muscle groups on performance of multiple degree-of-freedom input. *Proceedings of CHI'96*. 1996. ACM. pp. 308-315.
Keywords: *6DOF input, 3D user interfaces, 3D interaction, motor control, interactive rotation, clutching, Finger mouse, glove, experimental study*
Annotations: The paper evaluates two modes of manual control in 6DOF input. In the first mode the user can use fingers in 3D object rotation, while in the second fingers are excluded, forcing the user to use the whole hand for manipulating objects (which is exactly what happens in typical VR application when magnetic sensor is attached to the data glove). The experiments showed that device that utilized fingers allowed for significantly more effective 3D object manipulation. The author suggested that design of devices for 6DOF input should invite fingers participation in input control.