

Augmented Reality: The Interface is Everywhere

Ronald Azuma
Mark Billinghurst
Tobias Höllerer
Hirokazu Kato
Ivan Poupyrev
Dieter Schmalstieg

Course 27
Siggraph 2001

<http://www.hitl.washington.edu/ARtutorial/>

Augmented Reality: The Interface is Everywhere

Contents

Introduction

Tutorial Presentations

Ronald Azuma:	Introduction to Augmented Reality
Ronald Azuma:	Head Tracking for Augmented Reality
Ivan Poupyrev:	Interaction Techniques for Augmented Reality
Mark Billinghurst:	Collaborative Augmented Reality
Dieter Schmalstieg:	Heterogeneous Augmented Reality
Tobias Höllerer:	Mobile Augmented Reality
Hirokazu Kato:	Developing Applications with ARToolKit

Color Pages

Augmented Reality Bibliography

Selected Papers

- R. Azuma: A Survey of Augmented Reality
- R. Azuma, G. Bishop: Improving Static and Dynamic Registration in an Optical See-Through HMD
- S. You, U. Neuman, R. Azuma: Hybrid Inertial and Vision Tracking for Augmented Reality Registration
- M. Billinghurst, H. Kato, I. Poupyrev: Tangible Augmented Reality
- I. Poupyrev, et. al. Tiles: A Mixed Reality Authoring Interface.
- M. Billinghurst, H. Kato, I. Poupyrev: The MagicBook: A Transitional AR Interface
- D. Schmalstieg, et., al. : The *Studierstube* Augmented Reality Project
- D. Schmalstieg, A. Fuhrmann, G. Hesina: Bridging Multiple User Interface Dimensions with Augmented Reality
- A. Butz, et., al.: An Experimental Hybrid User Interface for Collaboration
- S. Feiner, B. MacIntyre, T. Höllerer, A. Webster: A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment
- H. Kato, M. Billinghurst: Marker Tracking and HMD Calibration for a Video-based Augmented Reality Conferencing System.

Augmented Reality: The Interface is Everywhere

Organizers: Mark Billinghurst, Dieter Schmalstieg

This course presents a thorough introduction to Augmented Reality (AR) including a review of AR technology, important research areas and cutting edge applications. Leading researchers will cover topics progressing from fundamental AR technology to advanced user interface techniques and applications. In addition to featuring hands-on demonstrations we will provide attendees with a general-purpose AR toolkit, equipping participants with the skills they need to start developing their own AR applications.

As computers become more and more invisible, Augmented Reality (the overlaying of virtual images on the real world) is becoming an increasingly important application area for computer graphics and user interface design. The user interface can literally be placed everywhere. This course is designed to provide attendees with the background, skills and software necessary to start developing AR applications. Attendees will be given a detailed introduction to AR technology with in-depth reviews of important research areas such as tracking and registration, interaction techniques, wearable AR systems and hybrid AR interfaces. They will also be able to try several AR demonstrations to experience the technology for themselves, and will be given a detailed introduction to ARToolKit, a software library that enables developers to easily build their own applications.

Alphabetical List of Presenters:

Ron Azuma	- azuma@HRL.com
Mark Billinghurst	- grof@hitl.washington.edu
Tobias Höllerer	- htobias@cs.columbia.edu
Hirokazu Kato	- kato@sys.im.hiroshima-cu.ac.jp
Ivan Poupyrev	- poup@csl.sony.co.jp
Dieter Schmalstieg	- dieter@cg.tuwien.ac.at

Course Presenter's Biographies:

Ronald Azuma

Ronald Azuma built the first motion-stabilized optical see-through Augmented Reality system. His work has been published in two SIGGRAPH papers and elsewhere, including a well-known survey paper of the field. Recently he has been pursuing accurate registration in outdoor environments. Ronald Azuma's current research interests are in the areas of Augmented Reality, virtual environments, 3-D interactive computer graphics and visualization. He is currently a Senior Research Staff Computer Scientist at HRL Laboratories in Malibu, California. Prior to joining HRL, he received a B.S. in Electrical Engineering / Computer Science, from UC Berkeley, and M.S. and Ph.D. degrees in Computer Science from UNC Chapel Hill

Mark Billinghurst

Mark Billinghurst is a final year PhD student at the Human Interface Technology Laboratory (HIT Lab) at the University of Washington, Seattle. He is active in several

research areas including augmented and virtual reality, conversational computer interfaces and speech and gesture recognition. His most recent work centers around using wearable computers and augmented reality to enhance face to face and remote conferencing. He is technical manager of the HIT Lab's wearable computing and augmented reality research projects and has collaborated on projects with the US Navy, ATR Research Labs in Japan, British Telecom and the MIT Media Laboratory. He has presented tutorials at the VRAIS 96, VRST 96, Visual 98 and HUC 99 conferences and has authored or co-authored more than 50 peer reviewed journal and conference papers.

Tobias Höllerer

Tobias Höllerer is a Ph.D. candidate and Graduate Research Assistant in the department of Computer Science at Columbia University. For the past five years he has been working with Professor Steven Feiner on Augmented Reality and 3D user interfaces. He is writing his Ph.D. thesis on user interfaces for mobile augmented reality systems (MARS). Tobias received M.Phil. and M.S. degrees from Columbia University and a Diploma in Computer Science from the Technical University Berlin. He spent a summer each at Microsoft Research and Xerox PARC, where he was working on 3D user interfaces and information visualization. Prior to his work at Columbia University he was doing research in scientific visualization and natural language programming. His main research interests lie in augmented reality, mobile and wearable computing, and adaptive 3D user interfaces.

Hirokazu Kato

Hirokazu Kato received the B.E., M.E. and Dr.Eng. degrees from Osaka University, Japan in 1986, 1988 and 1996 respectively. He joined the Department of Control Engineering at Osaka University, from 1989 to 1999. In 1998 he joined the Human Interface Technology Laboratory (HIT Lab) at the University of Washington as a visiting scholar and worked for the Shared Space project. Since 1999 he has been with the Department of Information Machines and Interfaces at Hiroshima City University, Japan, where he is currently an associate professor. He has been studying pattern recognition, image processing and computer vision. Also he has been interested in human computer interaction and computer mediated communication.

Ivan Poupyrev

Ivan Poupyrev is a Researcher in the Interaction Lab at the Sony CS Labs in Tokyo. Prior to joining Sony he spent two years at the MIC Labs, ATR International in Kyoto, where he was conducting research in augmented reality interfaces. While working on his doctorate degree in Computer Science he spent two and a half years at the Human Interface Technology Laboratory at the University of Washington as a Visiting Scientist designing and investigating 3D user interfaces for virtual reality and desktop 3D applications. His current research interests are in the intersection between advanced human-computer interfaces and computer graphics, including 3D interfaces, augmented reality interfaces, ubiquitous and wearable computing. The results of his research have been widely presented at major international conferences such as UIST, CHI, SIGGRAPH, EUROGRAPHICS, VRAIS and others. He co-authored a SIGGRAPH 2000 course "3D user interface design: Fundamental Techniques, Principle and Practice" and

received a M.S. in Applied Mathematics and Computer Science from Moscow Airspace University, Soviet Union.

Dieter Schmalstieg

Dieter Schmalstieg is a faculty member at Vienna University of Technology, Austria, where he leads the "Studierstube" research project on collaborative augmented reality. His current research interest are virtual environments, augmented reality, three-dimensional user interfaces, and distributed graphics. He currently leads research projects on mobile augmented reality and real-time visualization of urban environments, and is involved in the EC-funded Platform for Animation and Virtual Reality. Schmalstieg received an MSc (1993) and PhD (1997) degree from Vienna University of Technology. He is author and co-author of over 40 scientific publications, editorial advisory board member of computers & graphics, and organizer of the Eurographics workshop on virtual environments 1999.

Introduction to Augmented Reality

Ronald Azuma
Senior Research Staff Computer Scientist
HRL Laboratories, LLC
3011 Malibu Canyon Rd MS RL96
Malibu, CA 90265
azuma@HRL.com
<http://www.cs.unc.edu/~azuma/>



Definition of Augmented Reality (1)



- Virtual Environments (VE): Completely replaces the real world
- Augmented Reality (AR): User sees real environment; combines virtual with real
- **Supplements** reality, instead of completely replacing it
- Photorealism not necessarily a goal



Example AR image



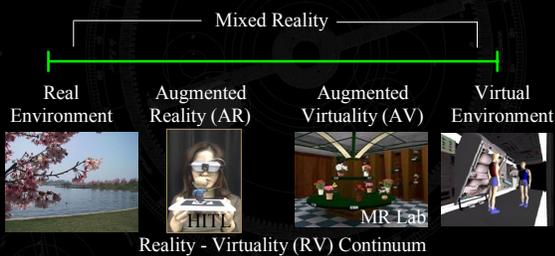
Definition of Augmented Reality (2)



- 1) Blends real and virtual, in real environment
- 2) Real-time interactive
- 3) Registered in 3-D
 - Applies to all senses (auditory, haptic?)
 - Not an HMD-specific definition
 - Includes idea of removing part of real environment (a.k.a. mediated or diminished reality)



Milgram's Reality-Virtuality continuum



Adapted from Milgram, Takemura, Utsumi, Kishino. Augmented Reality: A class of displays on the reality-virtuality continuum



Why are researchers interested?



- Enhance perception of and interaction with the real world
- Potential for productivity improvements in real-world tasks
- Relatively new field with many problems, but much progress has occurred recently



A Brief (and incomplete) History of AR (1)



- 1960's: Sutherland / Sproull's first HMD system was see-through



A Brief (and incomplete) History of AR (2)



- Early 1990's: Boeing coined the term "AR." Wire harness assembly application begun.
- Early to mid 1990's: UNC ultrasound visualization project
- 1994: Motion stabilized display [Azuma]
- 1994: Fiducial tracking in video see-through [Bajura / Neumann]



A Brief (and incomplete) History of AR (3)



- 1996: UNC hybrid magnetic-vision tracker (first compelling environment)
- 1998: Dedicated conferences begin
- Late 90's: Collaboration, outdoor, interaction
- Late 90's: Augmented sports broadcasts
- 1998 - 2001: Mixed Reality Systems Lab
- 2000: Custom see-through HMDs



Growth of field: conferences



New conferences dedicated to this topic:

- **International Symposium on Augmented Reality**
<http://www.Augmented-Reality.org/isar>
- **International Symposium on Mixed Reality**
<http://www.mr-system.co.jp/ismr>
- **Designing Augmented Reality Environments**



Growth of field: projects



- **Mixed Reality Systems Laboratory (Japan)**
http://www.mr-system.co.jp/index_e.shtml
- **Project ARVIKA (Germany)**
<http://www.arvika.de/www/e/miscel/sitemap.htm>
- **Ubicom Project (Delft University)**
<http://www.ubicom.tudelft.nl>



Some starting points



- **Jim Vallino's pointer page:**
<http://www.cs.rit.edu/~jrv/research/ar>
- **My survey paper**
Azuma, Ronald. A Survey of Augmented Reality.
Presence: Teleoperators and Virtual Environments 6, 4
(August 1997), 355-385.



More starting points



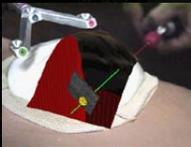
- Updated survey expected in Nov. 2001 IEEE Computer Graphics & Applications
- Book
Barfield and Caudell. Fundamentals of Wearable Computers and Augmented Reality. Lawrence Erlbaum Associates (2001). ISBN 0-8058-2901-6



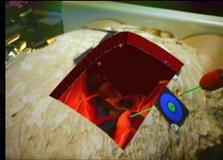
Applications: medical



- "X-ray vision" for surgeons
- Aid visualization, minimally-invasive operations. Training. MRI, CT data.
 - Ultrasound project, UNC Chapel Hill.



Courtesy
UNC
Chapel
Hill



Applications: complex machinery



- Instructions for assembly, maintenance and repair of complex equipment
 - Aircraft [Boeing]
 - Printers [Columbia]
 - Engines
 - Automobile assembly
 - and others...



Assembly and maintenance pictures (1)



Boeing wire harness assembly.
Adam Janin wearing HMD.
Courtesy David Mizell, Boeing



Courtesy Andrei State, UNC
Chapel Hill



Assembly and maintenance pictures (2)



© 1996 S. Feiner, B. MacIntyre, &
A. Webster, Columbia University



Columbia University



Eric Rose, et. al., ECRC



© 1993 S. Feiner, B. MacIntyre, &
D. Seligmann, Columbia University

Applications: annotating environment



- Public and private annotations
- Aid recognition, "extended memory"
 - Libraries, maps [Fitzmaurice93]
 - Windows [Columbia]
 - Mechanical parts [many places]
 - Reminder notes [Sony, MIT Media Lab]
 - Navigation and spatial information access



Annotation pictures



Columbia University




© 1993 S. Feiner, B. MacIntyre, M. Haupt, & E. Solomon, Columbia University

© 1997 S. Feiner, B. MacIntyre, T. Hollerer, & A. Webster, Columbia University




HRL

Application: broadcast augmentation



- Adding virtual content to live sports broadcasts
 - “First down” line in American football
 - Hockey puck trails, virtual advertisements
 - National flags in swimming lanes in 2000 Olympics
- Commercial application
 - Princeton Video Image is one company



Application: aircraft operations



- Helmet-mounted sights (short-range missiles)
- Virtual runway markers
 - Runway incursions are a leading cause of aircraft accidents.
 - T-NASA head up display for runway incursions
 - Enhanced view for low visibility situations



Application: collaboration



AR allows users to collaborate inside the same real environment

HIT Lab & ATR

Studierstube, Vienna University of Technology



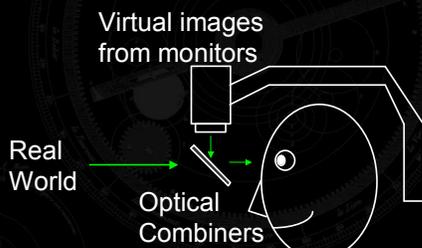
AR Systems Overview



- Blending: Optical vs. Video
- Focus, contrast, portability
- Sensing and bandwidth



Optical see-through head-mounted display



Examples of optical see-through HMDs

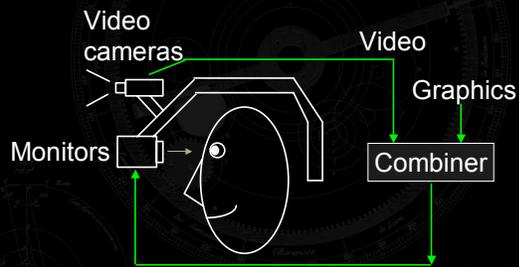


Sony Glasstron

Virtual Vision VCAP



Video see-through head-mounted display



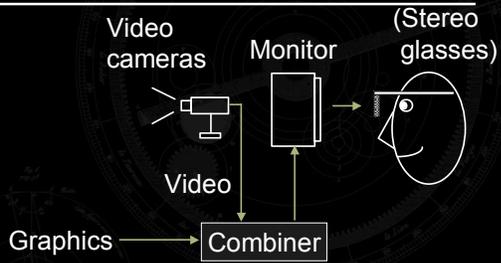
Example of video see-through HMD



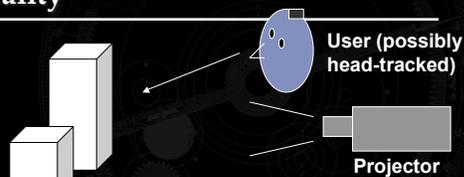
MR Laboratory's COASTAR HMD
(Co-Optical Axis See-Through Augmented Reality)
Parallax-free video see-through HMD



Video monitor Augmented Reality



Projector-based Augmented Reality

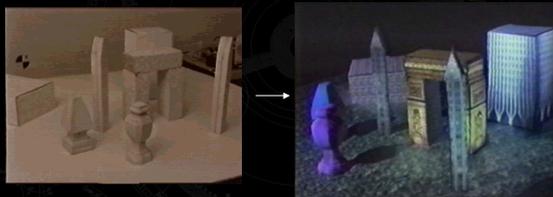


Real objects with retroreflective covering

Examples: Raskar, UNC Chapel Hill
Inami, Tachi Lab, U. Tokyo



Example of projector-based AR



Ramesh Raskar, UNC Chapel Hill



Optical strengths



- **Simpler (cheaper)**
- **Direct view of real world**
 - Full resolution, no time delay (for real world)
 - Safety
 - Lower distortion
- **No eye displacement (but COASTAR video see-through avoids this problem)**



Video strengths



- **True occlusion (but note Kiyokawa optical display that supports occlusion)**
- **Digitized image of real world**
 - Flexibility in composition
 - Matchable time delays
 - More registration, calibration strategies
- **Wide FOV is easier to support**



Optical vs. video summary



- **Both have proponents**
- **Video is more popular today?**
- **Depends on application?**
 - Manufacturing: optical is cheaper
 - Medical: video for calibration strategies



Focus and contrast



- **Focus**
 - Need to measure eye accommodation?
 - Autofocus video camera?
- **Contrast**
 - Desirable to match brightness
 - Real world has large dynamic range!
 - More difficult with optical?



Portability



- **VE: User stays in one place**
- **AR: User moves to task location**
 - Want to use in factories, outdoors, etc.
 - Less controlled environments
 - Very demanding of the technology



Requirements comparison vs. Virtual Environment systems



- **Rendering** ↓
- **Display (resolution, FOV, color)** ↓
- **Tracking and sensing** ↑
 - Greater bandwidth requirements (video, MRI data, range data, etc.)
 - Support occlusion, general environmental knowledge
 - A big problem for registration!



Upcoming course sections (1)

photos of people



- **Head Tracking for Augmented Reality**
 - 9:30 - 10:10 am
 - Ronald Azuma
 - The basic enabling technology
 - Registration approaches



Upcoming course sections (2)



- **Interaction Techniques for AR**
 - 10:30 - 11:15am
 - Ivan Poupyrev, Sony CSL
 - AR interface design
 - Novel input devices
 - AR widgets and elements
 - Evaluating interfaces



Upcoming course sections (3)



- **Collaborative Augmented Reality**
 - 11:15am - noon
 - Mark Billinghurst, Human Interface Technology Lab
 - Comparison against other collaboration
 - AR conferencing
 - Case and usability studies



Lunchtime Demos



- Noon - 1:50pm: Q&A, demos, lunch
- Demos run during the lunch break
 - The Magic Book
 - WearCom: wearable AR conferencing
 - ARstudy: a basic ARToolkit application



Upcoming course sections (4)



- Heterogeneous AR + Hybrid UI's
 - 1:50 - 2:40pm
 - Dieter Schmalstieg, Vienna U. Tech.
 - Alternative displays
 - Combinations with other UI metaphors
 - Sample applications



Upcoming course sections (5)



- Mobile AR
 - 2:40 - 3:30pm
 - Tobias Höllerer, Columbia University
 - Wearable and situated computing
 - Outdoor tracking
 - Interfaces and UI's



Upcoming course sections (6)



- **Developing applications with ARToolKit**

- 4:00 - 5:00pm
- Hirokazu Kato, Hiroshima City University
- Freely available toolkit for building applications
- Sample applications



Other current research directions (1)



- **Ease of setup and use**
 - Avoid need for expert user
 - Reduce calibration requirements
- **Human factors and perceptual studies**
 - Potential conflicts and optical illusions
 - Eye displacement in video see-through



Other current research directions (2)



- **Proven applications**
 - Need demonstrated performance improvements
- **Photorealistic rendering**
- **AR in other senses**
 - Recent haptic demo [Walairacht ISMR2001]
- **Social acceptance**
 - User perception of privacy, trust, and fashion!



Head Tracking for Augmented Reality

Ronald Azuma
Senior Research Staff Computer Scientist
HRL Laboratories, LLC
3011 Malibu Canyon Rd MS RL96
Malibu, CA 90265
azuma@HRL.com
<http://www.cs.unc.edu/~azuma/>



Related SIGGRAPH courses



- Course 8: "An Introduction to the Kalman Filter"
- Course 11: "Tracking: Beyond 15 Minutes of Thought"
- Gary Bishop and Greg Welch, UNC Chapel Hill
- Both courses occurred on Sunday...



Goals for this session



- Importance and difficulty of tracking
- Registration techniques
- Prediction
- Tracking technologies
- Fusing sensor information
- Research Directions



The importance of tracking



- Tracking is the basic enabling technology for Augmented Reality
- Without accurate tracking you can't generate the merged real-virtual environment
- Tracking is significantly more difficult in AR than in Virtual Environments

"Tracking is the stepchild that nobody talks about."
- Henry Sowizral, Dec 1994 Scientific American



The Registration Problem



- Virtual and Real must stay properly aligned
- If not:
 - Compromises illusion that the two coexist
 - Prevents acceptance of many serious applications
 - Do you want a surgeon cutting into you if the virtual cut-marks are misaligned?



Difficulty of registration



- Accurate registration is not trivial
 - Sensitivity of visual system (few mm, fraction of degree. Dime test.)
 - Many sources of error
- Demonstrate with ultrasound footage



Courtesy
UNC Chapel Hill



Types of registration



- Accuracy required depends on senses
- **Visual - Visual**
 - Errors are obvious. 0.5 minutes of arc
 - This is what we currently focus on for AR
- **Visual - Kinesthetic and Proprioceptive**
 - Main VE conflict, less obvious. Visual capture.
- **Visual - Auditory and Haptic**



Sources of registration errors



- **Static errors**
 - Optical distortions
 - Mechanical misalignments
 - Tracker errors
 - Incorrect viewing parameters
- **Dynamic errors**
 - System delays



Reducing static errors



- **Distortion compensation**
- **Manual adjustments**
- **View-based or direct measurements**
 - [Azuma94] [Caudell92] [Janin93] etc.
- **Camera calibration (video)**
 - [ARGOS94] [Bajura93] [Tuceryan95] etc.



Reducing dynamic errors (1)



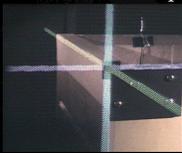
- Reduce system lag
 - [Olano95] [Wloka95a] [Regan SIGGRAPH99]
- Reduce apparent lag
 - Image deflection [Burbidge89] [Regan94] [So92] [Kijima ISMR 2001]
 - Image warping [Mark 3DI 97]



Reducing dynamic errors (2)



- Match input streams (video)
- Predict
 - [Azuma94] [Emura94] & others
 - Inertial sensors helpful



Azuma / Bishop, SIGGRAPH 94



The prediction problem



- Accurate prediction can be difficult
- "Like driving a car using only the rear view mirror"
 - Straight road = trivial
 - Curved road = maybe possible?
 - Right angle turns = forget it!



How well do existing predictors perform?



- Most predictor models are simple (e.g. constant acceleration)
- Empirically for HMD system, < 80 ms lag
 - Factor of 2-3 without inertial sensors
 - Factor of 5-10 with inertial sensors
- Can analyze specific linear predictors...
 - Azuma, Bishop [SIGGRAPH95]



How to improve prediction



- Better estimation and prediction
 - More sophisticated motion models
 - Bayesian and nonlinear approaches
 - Adaptive (since nonstationary)
 - Exploit correlations in motion data
 - Cleaner tracker outputs (since prediction blows up noise) with derivative measurements



Some cautionary notes on prediction



- Adaptive
 - Always switching "after the fact"
 - Doesn't tell you how to build models
 - Large errors if you choose incorrect model
- Complex (better models)
 - Compute time for predictor increases lag
 - Increasing lag makes problem harder
 - Harder problem -> complex predictor



Will the need for prediction disappear in the future?



- Computers are getting faster, correct?
- But faster user motion also...
 - Lighter HMDs and other equipment
 - Entertainment, high-performance situations
 - Hands and other body parts
- Prediction over shorter intervals?



Vision-based techniques (1)



- Digitized video allows “closed loop” approaches [Bajura 95]
- Difficult but not “AI complete” problem
- Popular due to accuracy. Made video see-through more common



Courtesy
UNC Chapel Hill



Vision-based techniques (2)



- Approaches used:
 - Fiducials in environment (LEDs, colored dots)
 - Template matching
 - Restricted environment with known objects
 - More sensors (e.g. laser rangefinder)
 - Keep user in the loop (manual identification)
- Requires compute power, I/O



Calibration-free approaches



- Registration generally involves significant calibration
- Rendering techniques that avoid certain calibration steps
 - Kutulakos, Vallino [IEEE TVCG vol 4 #1]
 - Seo, Hong [ISAR2000]



Registration: Current status



- Open-loop and closed-loop: precise in restricted cases
- Problems: limited range, motion, and environment
- Much work remains to be done!



Tracking technologies (as applied to AR)



- GPS
 - Regular ~30 meters, Differential ~3 meters
 - Carrier phase: centimeters but multipath and initialization problems
 - Line of sight, jammable
- Inertial and dead reckoning
 - Sourceless but drifts
 - Cost and size restrictions



Tracking Technologies (2)



- **Active sources**
 - Optical, magnetic, ultrasonic
 - Requires structured, controlled environment
 - Restricted range
 - Magnetic vulnerable to distortions
 - Ultrasonic: ambient temperature variations
 - Optical is often expensive



Tracking Technologies (3)



- **Scalable active trackers**
 - InterSense IS-900, 3rd Tech HiBall
- **Passive optical**
 - Line of sight, may require landmarks to work well. Can be brittle.
 - Computer vision is computationally-intensive



3rd Tech, Inc.



Tracking Technologies (4)



- **Electromagnetic compass, tilt sensors**
 - Passive and self-contained
 - Vulnerable to distortions
- **Mechanical**
 - Can be accurate but tethers user
- **Hybrid trackers**
 - Combines approaches to cover weaknesses
 - Yields the best results



TCM2



Fusion of Sensor Data (the software side of tracking)



- **Kalman filter for estimation**
 - Combines multiple measurements, when available, to reduce overall errors. Allows correlation among multiple signals.
 - Takes advantage of measured derivatives.
 - Empirically still works with nonideal models.
 - Linear approximation for nonlinear (EKF)
 - Computationally efficient



More notes on Kalman filters



- **Building a filter is easy**
 - But modeling problem and tuning filter is not.
- **Greg Welch site on Kalman filters**
 - <http://www.cs.unc.edu/~welch/kalman/>



SCAAT filter (1)



- **Welch and Bishop, SCAAT: Incremental Tracking with Incomplete Information (SIGGRAPH 97)**
- **SCAAT = Single Constraint at a Time**
- **Incorporates partial results *as they are measured* into the estimator**
- **E.g. one beacon or fiducial measurement**
- **Improves the solution**



SCAAT (2)



- Influential paper
- Benefits:
 - Filter update rate matches the fastest sampling rate in your system (can be kHz)
 - Greatly reduces temporal errors
 - Reduces computation time per iteration
 - Supports autocalibration



Research Directions in Tracking and Registration



- Hybrid tracking systems
 - Combine approaches, cover weaknesses
- Systems built for greater input variety and bandwidth
- Hybrid systems and techniques
 - e.g. use multiple registration techniques



Research Directions (2)



- True real-time systems
 - Must synchronize with the real world
 - Time becomes a first class citizen
 - Time critical rendering
- Perceptual and psychophysical studies: when is registration critical?
- Goal: Accurate tracking at long ranges, in unstructured environments





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

AR Interfaces: Why it is Important?

- **Designing AR system == interface design.**
- **Augmentation itself is not a final goal.**
- **Objective is a high quality user experience**
 - Appropriateness of AR interface to tasks and application requirements.
 - Ease of use and learning.
 - High performance and user satisfaction.
- **Different user interfaces require different AR tracking and display technology**

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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

AR technology is interface technology. The goal of designing and improving AR hardware and software, for example HMDs, tracking and registration techniques, and sensors, is to design augmented reality user interfaces that provide users with high-quality user experience and ease of use and learning. The choice of tracking techniques and display technologies depends on the interface model that is used in designing AR applications.

Lecture Overview

- **Introduction to AR interfaces**
 - What is AR interface and what is not?
 - Properties/challenges in AR interface design
- **AR 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 direction**

SIGGRAPH
2001 EXPLORING INTERACTION
AND DIGITAL IMAGERY

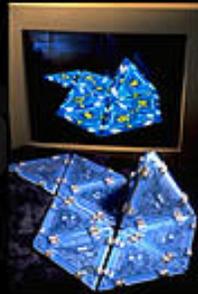
Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

This slide outlines the contents of this lecture.

What are AR interfaces?

- **Following Azuma definition of AR (1997)**

- a) combine real and virtual;
- b) interactive in real time;
- c) virtual objects are registered in 3D physical world



Triangles
Gorbet, et al. 1998



Environmental
displays

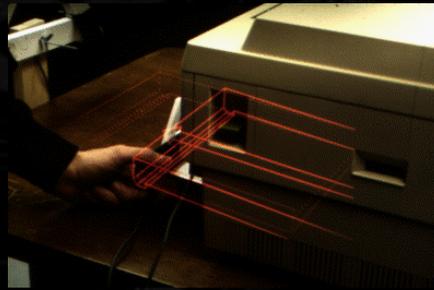
SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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

Before discussing AR interfaces in details, its useful to define them more precisely since there is a large variety of interfaces that can be called “AR”. in fact, we can talk about a continuum of AR interfaces, for example Milgram’s Mixed Reality continuum (1994). To focus this lecture, I will discuss only interfaces that follow Azuma’s definition. Therefore, I will not discuss systems 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.

Challenges in AR Interfaces

- **Traditionally purely visual augmentation rather than interaction (Ishii, 1997)**
- **Limitations of AR displays**
 - Precise, real time tracking, registration
 - Seamless interaction everywhere in 3D physical space
- **Limitations of controllers**
 - Precise, real time tracking, registration
 - Seamless interaction with both virtual and physical objects



KARMA, Feiner, et al. 1993

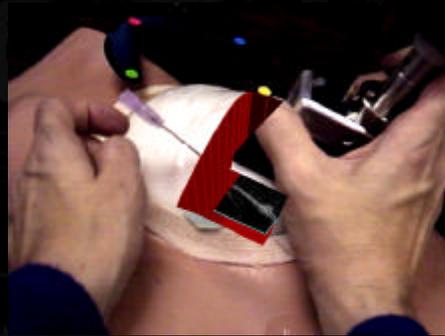
SIGGRAPH
2001 EXPLORING INTERACTION
AND DIGITAL MARKETS

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

AR has been traditionally used for visual augmentation, and its only been relatively recently that there's growing interest in AR interaction issues. 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
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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



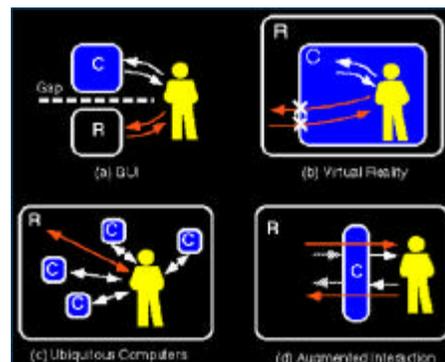
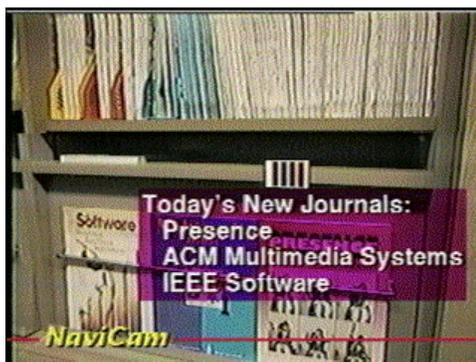
Rekimoto, et al. 1997

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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

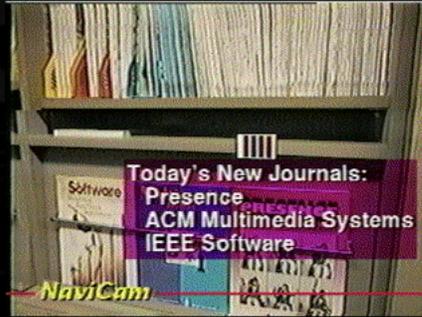
The data does not necessarily have to be 3D or modeled from the real world. Any information can be superimposed on the real world. Thus AR displays can present the data, e.g. text notes, voice or video annotations, etc, within a current real-world context. This approach was initially studied by Fitzmaurice (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
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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

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
EXPLORING INTERACTION
AND DIGITAL IMAGERY

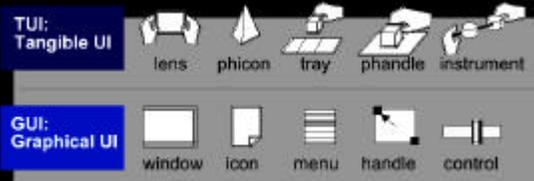
Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

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

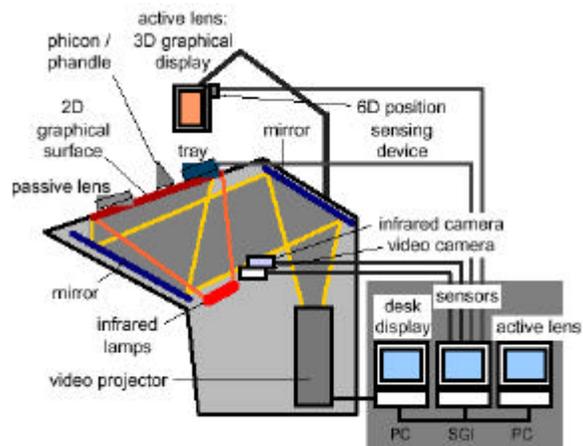


metaDesk. 1997



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
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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

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

SIGGRAPH
2001 EXPLORING INTERACTION AND DIGITAL WORLDS

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

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 EXPLORING INTERACTION
AND DIGITAL IMAGERY

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



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.

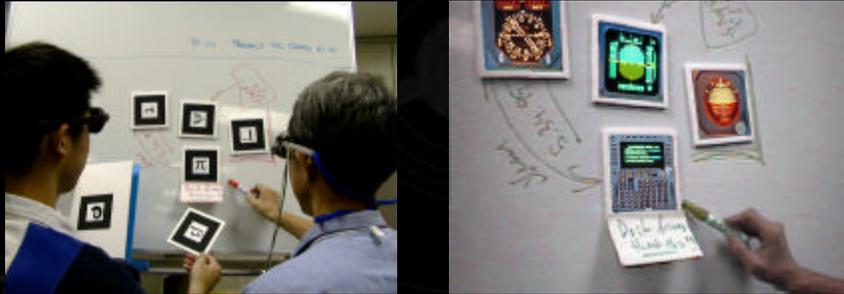
Menu tiles make up a book of the tiles attached to each page. This book works like a catalogue or a menu. As users flip through the pages, they can see the virtual objects attached to each page, choose the required instrument and then copy it from the book to any empty data tile.

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

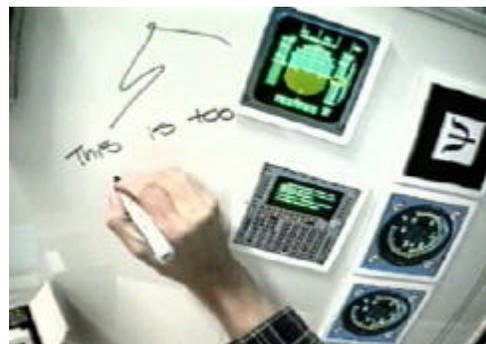


Data authoring in Tiles (Poupyrev, et al. 2001). Left, outside view of the system; right, view of the left participant.

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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 EXPLORING INTERACTION
AND DIGITAL IMAGERY

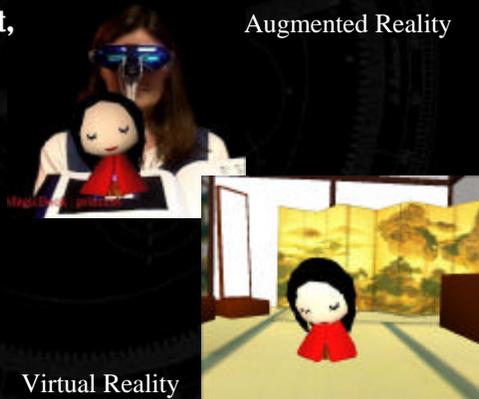
Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

The VOMAR project (Kato, et al. 2000) explored how a time-multiplexed tangible AR interface could be designed. In the project, a user uses a single input device 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 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): 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



Augmented Reality

Virtual Reality

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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

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
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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

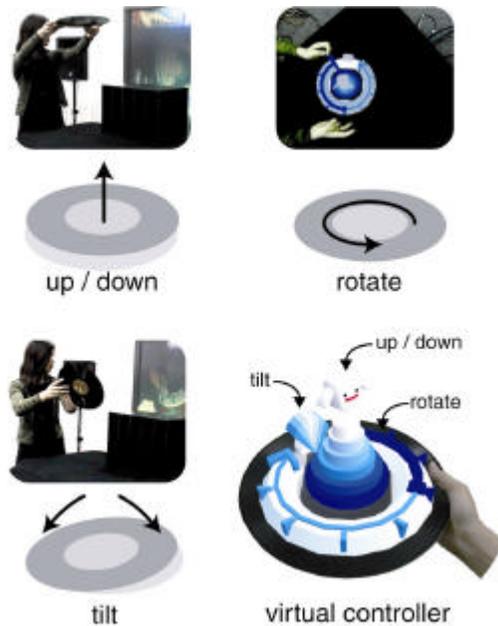


Augmented Groove, 2001



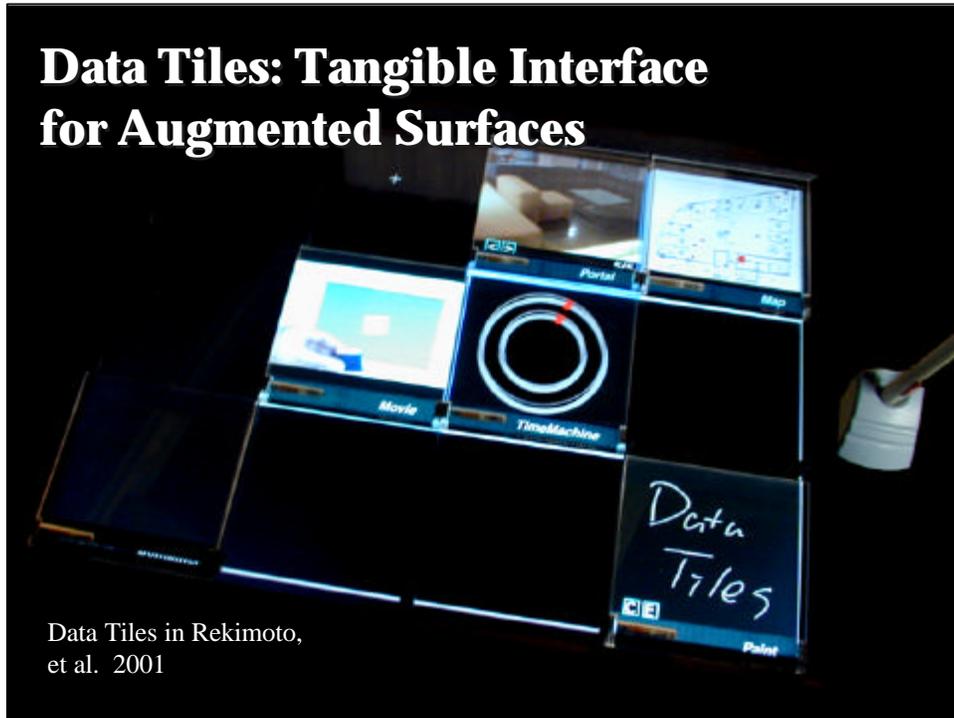
Copyright (c) Ivan Poupyrev, Interaction Lab Sony CSL 2001

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



Data Tiles in Rekimoto,
et al. 2001

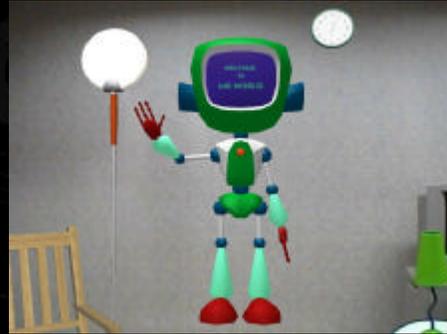
An interesting approach related to tangible AR was also designed and investigated in the DataTiles system by Rekimoto, et al. 2001. In this system, the user could arrange and interact with the virtual data by using transparent tiles that were placed on a flat sensor-enhanced display, through which the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

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 spatial 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
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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
EXPLORING INTERACTION
AND DIGITAL IMAGERY

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

The future is exciting.

Collaborative Augmented Reality

Mark Billinghurst
Human Interface Technology Laboratory
University of Washington, Seattle
grof@hitl.washington.edu

Collaboration and communication are two inherently human traits. For many thousands of years technologies have been developed to enable people to enhance their ability to connect with one another. From the development of writing to the printing press and telephone, a variety of inventions have enabled us to communicate with almost anyone, anywhere, anytime. The Information Age is no exception. As computers become more and more pervasive one of the pressing questions is how they too can be used to enhance face-to-face and remote collaboration.

Traditional teleconferencing and interfaces for computer supported collaborative work (CSCW) have many limitations. For remote conferencing, audio-only communication removes the visual cues vital for conversational turn taking, leading to increased interruptions and overlap, and difficulty in disambiguating between speakers and in determining other's willingness to interact. In video conferencing, some visual cues are present, however most non-verbal cues are not transmitted effectively, and the lack of spatial cues means that users often find it difficult to know when people are paying attention to them, to hold side conversations, and to establish eye contact. Collaborative virtual environments restore some of the spatial cues common in face-to-face conversation, but they require the user to enter a virtual world separate from their physical environment. Similarly, although the use of spatial cues and three-dimensional physical object manipulation are common in face-to-face communication, most CSCW systems do not provide collaborative virtual object viewing and manipulation in a face-to-face setting.

A relatively new technology, Augmented Reality (AR), can be used to overcome these limitations and support fundamentally different forms of collaboration. This is because of the unique characteristics of AR interfaces, including:

- Support of seamless interaction between real and virtual environments
- The ability to enhance reality and to create interfaces that go "beyond being there"
- The presence of spatial cues for face to face and remote collaboration
- Support of a tangible interface metaphor for manipulation of shared virtual objects
- The ability to transition smoothly between reality and virtuality

Seamless Interaction

In a face-to-face setting, when people talk to one another while collaborating on a real world task there is a dynamic and easy change of focus between the shared workspace and the speakers' interpersonal space. The shared workspace is the common task area between collaborators, while the interpersonal space is the common communications space. In a face-to-face meeting the shared workspace is often a subset of the interpersonal space, so participants move easily between spaces using a variety of non-verbal cues. For example, if architects are seated around a table with house plans on it, it is easy for them to look at the plans while simultaneously being aware of the conversational cues of the other people.

In most existing CSCW tools this is not the case. Current CSCW interfaces often introduce seams and discontinuities into the collaborative workspace. Ishii¹ defines a seam as a spatial, temporal or functional constraint that forces the user to shift among a variety of spaces or modes of operation. For example, the seam between computer word processing and traditional pen and paper makes it difficult to produce digital copies of handwritten documents without a translation step. Seams can be of two types:

- *Functional Seams*: Discontinuities between different functional workspaces, forcing the user to change modes of operation.
- *Cognitive Seams*: Discontinuities between existing and new work practices, forcing the user to learn new ways of working.

One of the most important functional seams is that between shared and interpersonal workspaces. For example, figure 1.0 shows a collaborative application that has a shared white board and a video window showing a remote collaborator. In this case the whiteboard and video windows are separated. This discontinuity prevents users who are looking at the shared white board from maintaining eye contact with their collaborators, an important non-verbal cue for conversation flow.

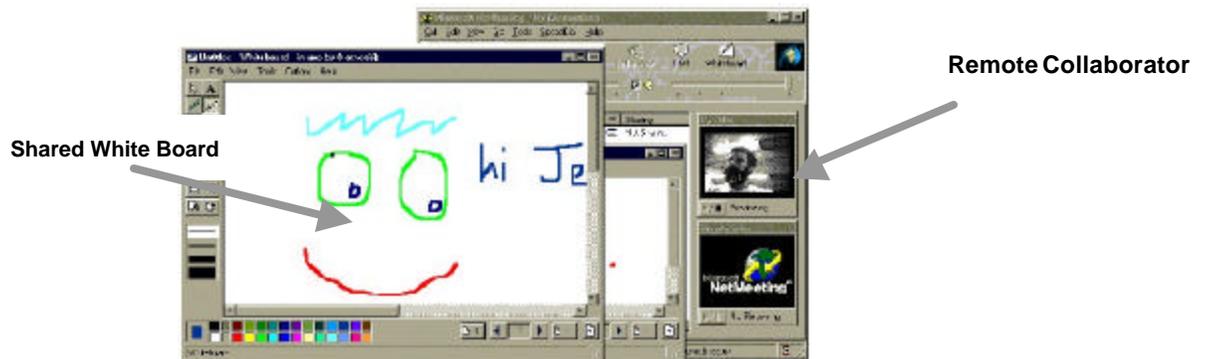


Figure 1.0. The seam between a shared whiteboard and video window.

A common cognitive seam is that between computer-based and traditional desktop tools. This seam causes the learning curve experienced by users who move from physical tools to their digital equivalents, such as the painter moving from oils to digital tools. CSCW tools are generally rejected when they force users to change the way they work; yet this is exactly what happens when collaborative interfaces make it difficult to use traditional tools in conjunction with the computer-based tools.

Seams in collaborative interfaces change the nature of collaboration and produces communication behaviors that are different from face-to-face conversation. So even with no video delay, video-mediated conversation doesn't produce the same conversational style as face-to-face interaction. This occurs because video cannot adequately convey the non-verbal signals so vital in face-to-face communication, introducing a seam between the participants. Thus, sharing the same physical space positively affects conversation in ways that is difficult to duplicate by remote means.

Collaborative Augmented Reality interfaces provide a new type of medium that can be used to reduce or remove the seams present in other collaborative interfaces. In an Augmented Reality interface it is possible to view shared virtual objects at the same

¹ Ishii, H., Kobayashi, M., Arita, K., Iterative Design of Seamless Collaboration Media. *Communications of the ACM*, Vol 37, No. 8, August 1994, pp. 83-97.

time as face-to-face collaborators, supporting the same type of dynamic interchange and range of conversational cues as in normal face-to-face conversation. AR also reduces cognitive seams by allowing people to use traditional physical tools to interact with virtual information.

Beyond Being There

Removing the seams in a collaborative interface is not enough. As Hollan and Stornetta² point out, CSCW interfaces may not be used if they merely provide the same experience as face-to-face communication; they must enable users to go “beyond being there” and enhance the collaborative experience. When this is not the case, users will often stop using the interface or use it differently than what it was intended for.

The motivation for going “beyond being there” can be found by considering past approaches to CSCW. Traditional CSCW research attempts to use computer and audio-visual equipment to provide a sense of remote presence. Measures of social presence and information richness have been developed to characterize how closely CSCW tools capture the essence of face-to-face communication. The hope is that collaborative interfaces will eventually be indistinguishable from actually being there.

This may be the wrong approach. Considering face-to-face interaction as a specific type of communications medium, it becomes apparent that this requires one medium to adapt to another, pitting the strengths of face-to-face collaboration against other interfaces. Mechanisms that are effective in face-to-face interactions may be awkward if they are replicated in an electronic medium. Rather than using new media to imitate face-to-face collaboration, researchers should be considering what new attributes the media can offer that satisfy the needs of communication so well that people will use it regardless of physical proximity.

Collaborative AR involves the addition of virtual images to real-world face-to-face or remote collaborations. Thus there is considerable potential for developing interfaces that go “beyond being there” that will be used regardless of physical proximity. For example, in a face-to-face meeting, virtual objects can appear between the participants that can interact with as easily as physical objects. In a remote collaboration one user could add virtual annotations into another’s field of view showing how to perform a real world task, or even “see” through their eyes, perceiving a remote location as if they were really there.

Spatial Cues

In face-to-face collaboration spatial cues are used to facilitate the collaborative process. During conversation the position and orientation of the participants, and their gaze and gestures are among the spatial cues used, while the spatial relationships between participants and physical objects, or between objects themselves are also important. These cues affect the collaborative flow, for example gaze is used to mediate turn-taking in conversation.

In traditional CSCW interfaces it is difficult to preserve these spatial cues. Multiparty video conferencing typically presents users in separate video windows on a two dimensional screen, removing the possibility of making eye-contact between

² Hollan, J., Stornetta, S. Beyond Being There. In *Proceedings of CHI '92*, 1992, New York: ACM Press, pp.119-125.

individual participants, or using body positioning and orientation to convey non-verbal cues. In a face-to-face collaboration CSCW tools often display digital content on a single two-dimensional screen, reducing three-dimensional virtual objects to a two dimensional projection and removing all but the simplest spatial cues between objects.

In contrast AR can be used to create three-dimensional virtual objects that appear as real as physical objects and inhabit the same space as real world physical objects. An AR interface can support the same spatial manipulations that people use with real objects and capture the same spatial relationships. It is even possible to use AR technology to bring remote users into a real environment a life-sized, three dimensional projections, facilitating the same spatial cues as used in normal face to face collaboration.

Tangible Interfaces

A fourth reason why AR technology can be used to create new types of collaborative interfaces is the support for a tangible interface metaphor. In face-to-face meetings physical objects or props are commonly used to convey collaborative information. In a collaborative setting speakers use the resources of the physical world to establish a socially shared meaning. 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 recent years, there has been movement to use physical objects as new input devices for computer interfaces. Ishii³ has coined the term “Tangible User Interface” or TUI, which has the goal of coupling digital information to everyday physical objects and environments. Augmented reality encapsulates the TUI concept and moves beyond it by allowing virtual enhancements to the physical interface objects. The physical objects can still be used to support collaboration, but they can also 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. An augmented physical object can still serve as the focus of a collaborative meeting, but now the semantics of the object can be exactly represented using attached virtual imagery.

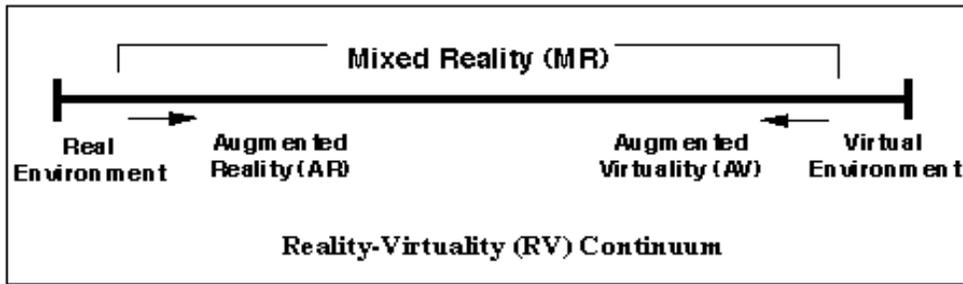
Transitional Interfaces

A final reason why AR techniques can be used to create compelling collaborative interfaces is that Augmented Reality can transport users smoothly between Reality and Virtual Reality.

Many computer interfaces have been developed which explore collaboration in a purely physical setting, in an AR setting, or an immersive VR environment. Milgram's taxonomy⁴ places these interfaces along a Reality-Virtuality continuum. Moving from left to right the amount of virtual imagery increases and the connection with reality weakens. However, collaborative interfaces typically do not allow people to move easily along this continuum and occupy discrete points in Milgram's Taxonomy.

³ 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.

⁴ Milgram, P., Kishino, F. A Taxonomy of Mixed Reality Visual Displays. IECCE Trans. on Information and Systems (Special Issue on Networked Reality), vol. E77 -D, no. 12, pp.1321-1329, 1994.



Milgram's Reality-Virtuality Continuum

Human activity often cannot be broken into discrete components and for many tasks users may prefer to be able to move seamlessly along the Reality-Virtuality continuum. This is particularly true when interacting with three-dimensional graphical content, either creating virtual models or viewing them. For example people using 3D model building software will often stop and draw away from the computer screen to sketch ideas on real paper. If a person wants to experience a virtual scene from different scales then immersive VR may be ideal, but if they want to have a face-to-face discussion while viewing the virtual scene an AR interface may be best.

In an Augmented Reality interface the amount of the real world that is enhanced or replaced by virtual imagery is entirely determined by the AR application. Thus AR techniques can be used in a transitional interface to move the user from a purely real to a purely virtual environment. This can also be used in a collaborative setting to support multi-scale collaboration; meaning that users immersed in the virtual scene (seeing an egocentric view) can still collaborate with users watching them from an exocentric viewpoint in the AR interface.

Collaborative Augmented Reality

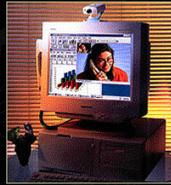
Mark Billinghurst
Human Interface Technology Lab.
University of Washington
Box 352-142
Seattle, WA 98195, USA
grof@hitl.washington.edu
<http://www.hitl.washington.edu/>



Today's Technology

Video Conferencing

- lack of spatial cues
- limited participants
- 2D collaboration



Collaborative VEs

- separation from real world
- reduced conversational cues



Beyond Video Conferencing

2D Interface onto 3D

- VRML

Projection Screen

- CAVE, WorkBench

Volumetric Display

- scanning laser

Virtual Reality

- natural spatial cues



Beyond Virtual Reality

Immersive Virtual Reality

- separates from real world
- reduces conversational cues

Lessons from CSCW

- Seamless
- Enhance Reality



SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Seamless CSCW

Seam (Ishii et. al.)

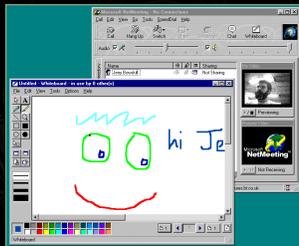
- spatial, temporal, functional discontinuity

Types of Seams

- Functional
 - between different functional workspaces
- Cognitive
 - between different work practices

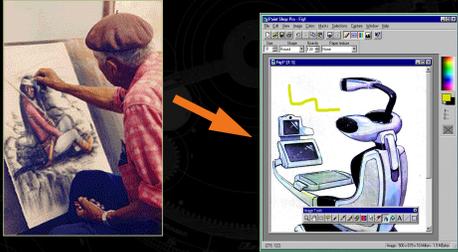
SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Functional Seams



SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Cognitive Seams



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Effect of Seams

Functional Seams:

- Mediated differs from F-to-F Conversation
 - Loss of Gaze Information
 - Degradation of Non-Verbal Cues

Cognitive Seams:

- Learning Curve Effects
- User Frustration

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Collaborative Augmented Reality

Facilitates seamless Collaboration

- Merges task space and communication space
 - No Functional Seams
- Blends Reality and Virtual Reality
 - No Cognitive Seams

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Collaborative AR Systems

Face to Face Conferencing

- Studierstube
- Shared Space

Remote Conferencing

- WearCom
- AR Conferencing Space

Transitional

- MagicBook

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Face to Face Conferencing

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Studierstube (Schmalstieg et. al.)

- "Studierstube" = "study room"
- collaborative AR
- virtual objects,
natural communication
- independent views of the data
 - POV, layers, annotations
- new forms of 3D interaction
 - Pen, PIP, tangible input devices



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Studierstube Video

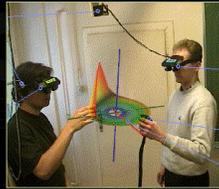


Studierstube Features

Seamless Interaction
Natural Communication

Attributes:

- Virtuality
- Augmentation
- Cooperation
- Independence
- Individuality



Merges Task and Communication Space



Shared Space (Siggraph 99)

Goal

- create compelling collaborative AR interface usable by novices

Exhibit content

- matching card game
- face to face collaboration
- physical objects
 - 5x7" cards
- built on VRML parser



Physical Interactions



Explored interactions between physical props

- relative position triggered animation

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Shared Space Video



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Results

2,500 - 3,000 users

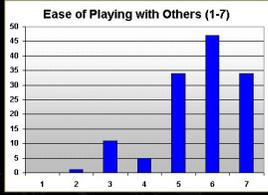
Observations

- no problems with the interface
 - only needed basic instructions
- physical objects easy to manipulate
- spontaneous collaboration

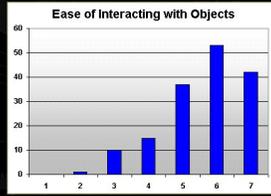


SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

User Results



1 = not very easy
7 = very easy



SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

User Feedback

Subject survey (157 people)

- Easy to play with other people
- Easy to interact with virtual objects

Best features

- interactivity, how fun it was, ease of use

Improvements

- reduce lag, improve image quality, better HMD

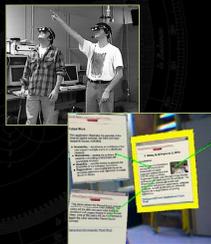
Applications

- games, education

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Related Work

- TransVision (Rekimoto)
- AR² Hockey (MRSL)
- RV Border Guards (MRSL)
- Collaborative Web Space (Billinghurst)



SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Remote Collaboration

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

WearCom

A cell phone with Intel inside ?

- What is it good for ?
- Is it better than a conference phone call ?

Wearable Conferencing Space

- wearable computer
- see-through HMD
- wireless connectivity



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

A Wearable Conferencing Space

Features

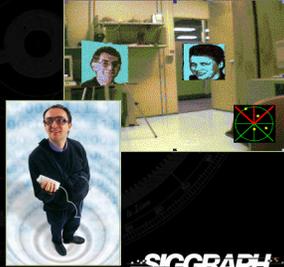
- Mobile video conferencing
- Full size images
- Spatial audio/visual cues
- Dozens of simultaneous users



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

WearCom Prototype

- Internet Telephony
- Spatial Audio/Visuals
- See-through HMD
- Inertial Head Tracking
- Wireless Internet
- Wearable Computer
- Static Images



SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Show WearCom Demo



SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Pilot User Study

Can Spatial Cues Aid Comprehension?

Task

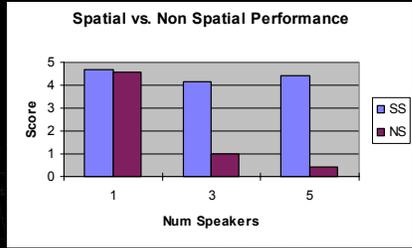
- recognize words in spoken phrases
- "My favorite food is _____. I like it very much"

Conditions

- Number of speakers
 - 1,3,5 simultaneous speakers
- Spatial/Non Spatial Audio
- Different Visual Cues

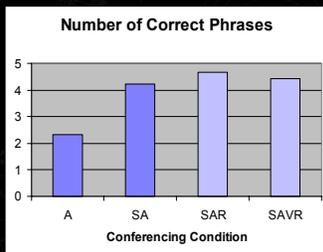
SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Spatial Sound Results



SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

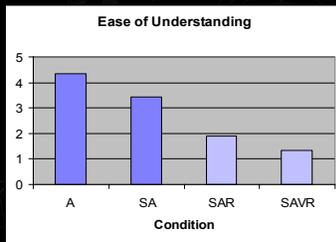
Spatial Visuals



A = audio only
SA = spatial audio
SAR = radar display
SAVR = virtual avatars

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Subjective Rankings



A = audio only
SA = spatial audio
SAR = radar display
SAVR = virtual avatars

1 = very easy, 5 = not very easy

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Augmented Reality Conferencing

Moves conferencing from the desktop to the workspace

SIGGRAPH
2001
EXPLORE INTERACTION AND DIGITAL IMAGES

Show AR Conferencing Video

SIGGRAPH
2001
EXPLORE INTERACTION AND DIGITAL IMAGES

Pilot Study

How does AR conferencing differ ?

Task

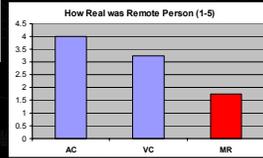
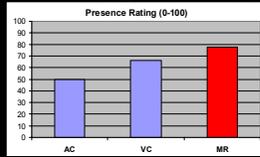
- discussing images
- 12 pairs of subjects

Conditions

- audio only (AC)
- video conferencing (VC)
- mixed reality conferencing (MR)

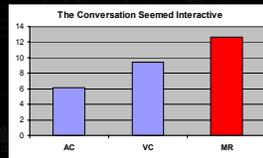
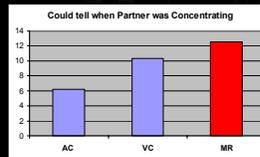
SIGGRAPH
2001
EXPLORE INTERACTION AND DIGITAL IMAGES

Presence



SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Communication



SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Results

Subjective Results

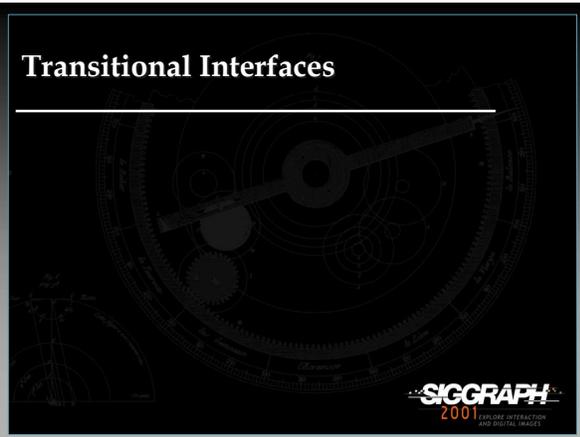
- AR conferencing can increase presence
- AR conferencing can improve communication
- AR more difficult to use than audio/video
 - Difficult to see everything
 - Communication asymmetries

Confounding Factors

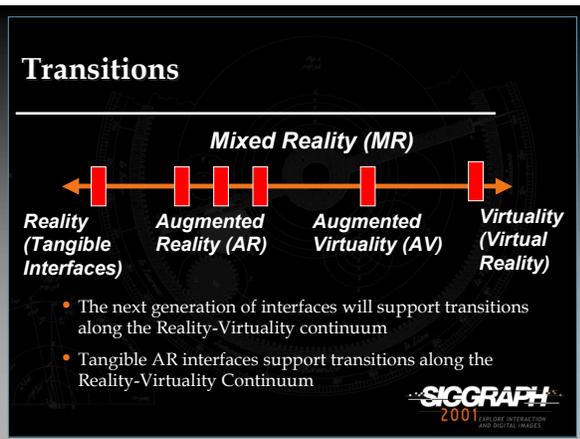
- task - not strictly conversation
- HMD - limited field of view, resolution
- only two users

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Transitional Interfaces



Transitions



Need for Transitional Interfaces

Interfaces of the future will need to support transitions along the RV continuum

Augmented Reality is preferred for:

- co-located collaboration

Immersive Virtual Reality is preferred for:

- experiencing world immersively (egocentric)
- sharing views
- remote collaboration



The MagicBook

Supporting collaboration in physical, AR and immersive VR setting

Book metaphor



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

MagicBook Video



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Features

Seamless transition between Reality and Virtuality

- Reliance on real decreases as virtual increases

Supports egocentric and exocentric views

- User can pick appropriate view

Computer becomes invisible

- Consistent interface metaphors
- Virtual content seems real

Supports collaboration

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Collaboration in the MagicBook



Egocentric



Exocentric

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Collaboration

Collaboration on multiple levels:

- Physical Object
- AR Object
- Immersive Virtual Space

Egocentric + exocentric collaboration

- multiple multi-scale users

Independent Views

- Privacy, role division, scalability

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Interface Components



Handheld Display



Physical Book



SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Technology

Reality

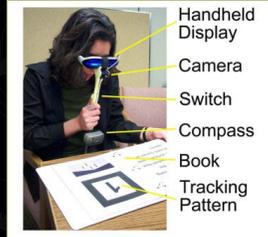
- No technology

Augmented Reality

- Camera - tracking
- Switch - fly in

Virtual Reality

- Compass - tracking
- Press pad - move
- Switch - fly out



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

User Feedback

"I think this is a great step towards immersive imagination."

"Great idea! I liked the handheld device."

Likes:

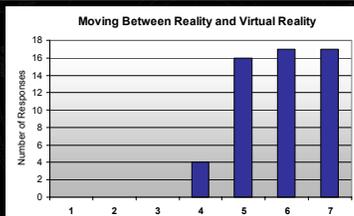
- The Augmented Reality book scenes (8)
- How innovative and cool it was (8)
- The camera tracking (6)

DisLikes:

- The realism of the graphics content (16)
- Movement through the Virtual Space (8)
- The image quality (5)

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Ease of Transition



How easily could you move between the real and virtual worlds?
(1 = not very easily, 7 = very easily)

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Related Work

Transitional Interfaces

- Kiyokawa - AR<-> VR
- Benford et. al. - Physical <-> VR
- Conway - WIM - VR navigation



Tangible Interfaces / Books

- Ishii, Lindemann, Hinckley, Stifelman

Multi-scale Collaboration

- Kiyokawa, Leigh



Lessons Learned

Face to face collaboration

- AR preferred over immersive VR
- AR facilitates seamless/natural communication

Remote Collaboration

- AR spatial cues can enhance communication
- AR conferencing improves video conferencing
- AR supports transitional interfaces



Areas for Future Work

Wearable collaborative AR system

- opportunistic collaboration
- just in time training

Communication Asymmetries

- interface, expertise, roles

Usability Studies

- multi-user AR systems
- communication tasks



Heterogeneous User Interfaces

Dieter Schmalstieg
Vienna University of Technology, Austria
dieter@cg.tuwien.ac.at

Augmented Reality (AR) is the combination of a user's perception of the real world with computer generated images. Per definition, it is a heterogeneous technique. It is noteworthy that AR is not a single method, but there are a number of possible combinations of real and virtual. Milgram has describes this situation as a *virtuality continuum*. For example, a see-through head-mounted display is used to create traditional *augmented reality*, while the combination of transparent props with back-projection can be characterized as *augmented virtuality*.

When designing a new user interface, one must make choices along a number of other continuums:

- **Display continuum:** In AR, users carry their own displays, while ubiquitous computing is based on the idea that display interfaces are embedded in large numbers everywhere. Both extremes allow to make computer generated information location-independent
- **User continuum:** A system may support only a single user, but networked systems become more interesting if they support multiple users, either co-located or remote. In the extreme, large users groups can be supported.
- **Application continuum:** A single application and its environment can be optimized for maximum performance and experience. This is the way currently taken by computer games. In contrast, concurrent execution of several applications allows to construct more interesting work environments. Ubiquitous computing is based on the idea of lots of applications embedded into the environment.

Many combinations from spots along these continua make sense for constructing interesting new user interfaces. Some prototypes also combine different user interface paradigms, such as AR, desktop computing and tangible user interfaces.

Finally, recent work has focused on creating heterogeneous user interface management systems that allow a user interface designer to mix and match user interface elements from different continua and styles as appropriate. The result is a heterogeneous work environment that surrounds user(s) and gives them access to the computer-mediated resources.

A key issue here is how to bridge the physical as well as conceptual space between the individual interaction platforms that may be very different in appearance, intuition and ergonomics. Several techniques, such as using props or surfaces as mediators have been explored. Other relevant issues include interaction with real and virtual objects, and privacy management for multi-user situations.



Heterogeneous Augmented Reality

Dieter Schmalstieg
Vienna University of Technology
Austria





Heterogeneous user interfaces

- AR combines real + virtual
-> implicitly heterogeneous
- AR is a tool, not an end to itself
- There are multiple flavors of AR





Milgram's continuum revisited

My desk	MagicBook	Transparent Props	"Gothic" RPG
			
	[Billinghurst2001]	[Schmalstieg99]	
Reality	Augmented Reality (AR)	Augmented Virtuality (AV)	Virtuality

All these options make sense for certain applications



Display continuum

Ultra-sound biopsy [State96]

Automated Teller Machine [Kaufman2000]

“Classic” Augmented Reality

- Users carry their computers
- See-through head mounted display, hand-held display

Ubiquitous computing

- Computers are embedded in environment
- Access to networked resources
- Active Surfaces

TU WIEN

SIGGRAPH 2001 EXPLORE INTERACTION AND DIGITAL IMAGES

User continuum

3D Browsing

3D Teachware [Kaufman2000]

Internet Games

Single user

Collaborating users, co-located

Collaborating users, remote

TU WIEN

SIGGRAPH 2001 EXPLORE INTERACTION AND DIGITAL IMAGES

Application continuum

Computer Game

My PC desktop

Single-tasking
Dedicated environment

Multi-purpose
Multi-tasking

TU WIEN

SIGGRAPH 2001 EXPLORE INTERACTION AND DIGITAL IMAGES

Combinations make sense



E.g., *Magic Book* uses 2 positions along Real-Virtual Continuum



Augmented Reality and Immersive Virtual Reality



Some other combinations



- DataTiles: Tangible + GUI
- VOMAR: Tangible + AR
- MARS: Indoor + outdoor AR
- Personal Interaction Panel: AR + GUI



DataTiles: Tangible + GUI



[Rekimoto2001]



VOMAR: Tangible + AR

TU
WIEN

[Kato2000]

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Personal Interaction Panel: AR+GUI

AR + GUI [Zsalavári97]

- pen and pad props
- two-handed interaction
- tactile feedback
- general and versatile tools
- natural embedding of 2D in 3D
- simple, cheap hardware

TU
WIEN

2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

MARS: GUI + Indoor + Outdoor AR

TU
WIEN

[Höllner99] Campus information system

GUI

Indoor AR

Outdoor AR

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Heterogeneous environment management



- **Idea: create a software framework to manage interactions**
 - of multiple users
 - in a distributed environment
 - using a variety of input/output platforms
- **Infrastructure used both by**
 - Users - ubiquitous computing
 - Applications - present a hybrid UI

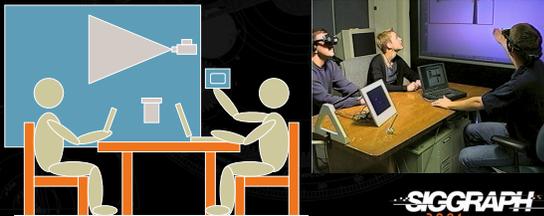


Motivation: Augmented Conferencing



EMMIE [Butz99]

Conferencing assisted by multiple computing devices



Design Ideas



Use the most appropriate tools for any given task

- Manipulate 2D text or images on a 2D PC or laptop
- Manipulate 3D objects in 3D space

Use the most appropriate displays

- size, resolution, stereopsis
- privacy vs sharing



Bridging Space (1)

Office of the Future [Raskar98]

- office environment augmented with embedded front projection
- 3D video conferencing



Bridging Space (2)

Emmie [Butz99]

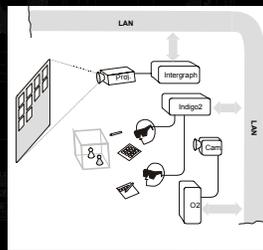
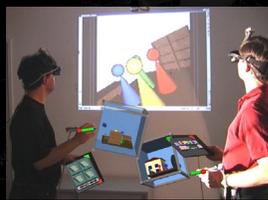
- Shared virtual "ether" metaphor
- Incorporate existing standard applications



Bridging Space (3)

Studierstube (V2.0) [Schmalstieg2000]

- Similar multi-display AR
- Mixed view applications
- Example: Storyboard design



Bridging Space (4)
mediaBlocks [Ullmer98]

- Carry "data containers" across physical space

white board
 printer
 browser

TU WIEN
 SIGGRAPH 2001 EXPLORE INTERACTION AND DIGITAL IMAGES

Bridging Space (5)
Active Surfaces
 [Rekimoto99]

- Space between objects bridged by display surface

TU WIEN
 SIGGRAPH 2001 EXPLORE INTERACTION AND DIGITAL IMAGES

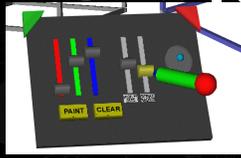
Bridging Space (6)
Multi-computer direct interaction
 [Rekimoto97]

[Rekimoto98]
 "pick-and-drop"

TU WIEN
 SIGGRAPH 2001 EXPLORE INTERACTION AND DIGITAL IMAGES

Analogy to Desktop Metaphor (1)

- Need tools for information manipulation
- Pointing device, menus, widgets
 - location or prop bound
- Direct manipulation, drag & drop in 3D



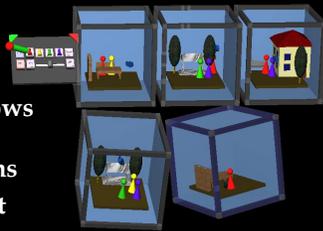
Pen & pad widgets

3D pointer, 2D laptop



Analogy to Desktop Metaphor (2)

- Multi-tasking
- 3D icons/windows representing data/applications
- multi-document interface



Multiple 3D document containers



Beyond the Desktop Metaphor

- Managing 3D space and multiple 2D/3D Displays
- Drag and drop between dimensionalities
- Multiple users -> privacy management
- Reaction to dynamic changes



Interaction with the virtual world



- Placing and manipulation of virtual objects
- Applications in the virtual
- 3D menus and widgets



[Butz99]



Interaction with the real world



- Placing and manipulating of tracked real objects
 - such as displays and computers
- Using interaction devices provided by participating computers
 - keyboards
 - 2D mice
 - pens
 - displays
- Using standard software on the various machines



Heterogeneous Interaction



Transition between Dimensions

- move objects between 2D and 3D displays

Hand-held displays as „windows“ to 3D

- visualize 3D without HMD



Privacy Management

TU
WIEN

- Select what information should be shared
- Simple and efficient
- Controlled by user
 - e.g. private notes, annotations
- Controlled by application
 - e.g. education, games

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Privacy Management (1)

TU
WIEN

Information layers
[Zsalavári98]



User specific textures on playtiles in Mahjongg game



SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Privacy Management (2)

TU
WIEN

Vampire mirrors

- show what others can see
- hybrid interaction technique



Privacy lamps

- emit a beam of privacy
- virtual interaction technique



[Butz99]

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Locales (1)

TU
WIEN

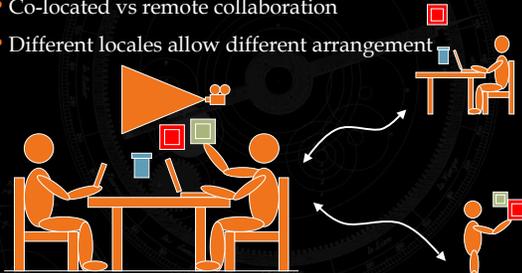
- Locomes allow "geometric privacy"
- 3D Ether: one world coordinate system
 - Privacy: need not share all information
 - Locomes: need not share all information arrangement
- Independent arrangement of information
 - Place application objects in different places
 - Different displays may imply different locales

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Locales (2)

TU
WIEN

- Co-located vs remote collaboration
- Different locales allow different arrangement



SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Locales (3)

TU
WIEN

Co-located replication in multiple locales

- Blur boundaries between co-located & remote
- E.g. mission control, teaching, presentation to large audiences

Consider presentation situation

- Interaction at arm's reach
- large screen presentation

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Wrap-up



- Many UI paradigms combinations make sense
 - AR, Desktop, Tangible, Immersive...
- Choose from several UI dimensions
 - real<->virtual, # of displays, users, applications...
- Heterogeneous UI requires
 - Space bridging metaphor
 - mixed real-virtual interaction metaphors
 - Multi-user, privacy, space (locale) management



Future Directions



- Better integration of 3D and ubiquitous computing
 - less experimental, more plug & play
 - better ergonomics
- Make everything mobile
 - > hear what Tobias Höllerer will tell you!



Heterogeneous Augmented Reality



- Thank you! -

Dieter Schmalstieg
Vienna University of Technology
Austria



Mobile Augmented Reality

Tobias Höllerer
Computer Graphics and User Interfaces Lab
Department of Computer Science
Columbia University
New York, NY

htobias@cs.columbia.edu



Mobile Augmented Reality



CO & UI Laboratory
Columbia University



Indoor and Outdoor
Mobile Augmented Reality Systems
(MARS)



Mobile Augmented Reality

Lecture Overview



CO & UI Laboratory
Columbia University

- Introduction to mobile AR
- Basics & Requirements
 - Hardware requirements
 - Tracking
 - Environmental modeling
- Mobile AR Systems
 - Existing systems in comparison
 - UI case study
- UI considerations and research topics



Mobile AR – Motivation



Mobile, wearable computing opens up new possibilities

- location-aware/situated computing

Now, the interface is truly everywhere

- AR is a powerful UI for this type of computing



Mobile AR – Motivation



Mobile AR Applications:

- Navigational aids
- Communication aids
- Personal situated information DB
- General UI for appliances
- Tourism
- Journalism
- Maintenance and construction
- Military training and warfighting



Mobile AR – Background



Post-WIMP interfaces:



Mobile AR – Background

Steps Toward Wearable Computing



Computer Form Factor	User Relationship
Room	Submit
Wall	Share
Desk	Sit at
Box	
Laptop	... and carry before/after
Palmtop	Hold
Clipboard	Wear



Implications of Wearability

(after S. Mann, B. Rhodes, T. Starner)



- Mobility**
 - usable/used indoors and outdoors
- Intimacy**
 - sense the wearer's body, communicate privately
- Context sensitivity**
 - take into account changing environment
- Constancy**
 - Permeation of UI into wearer's life



Mobile AR – Background

Situated Computing



Ubiquitous computing (Weiser '89)

PARCTab (1993)

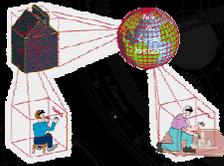
Hull et al. ('97) state that

“situated computing concerns the ability of computing devices to detect, interpret, and respond to aspects of the user's local environment”



Mobile AR – Background

WorldBoard



1990s: many researchers started to co-locate information with physical space

J. Spohrer 1996: What comes after the World Wide Web?

Information in place.
The world as a repository of information. (Imagined as a service Apple Computer, Inc. would provide.)



What is Mobile AR?

Ways of augmenting a mobile user's environment



- wearable display, no tracking whatsoever
- body-stabilized wearable display (orientation tracking only)
- location-dependent audio augmentation (with or without spatialized audio)
- location-dependent screen-stabilized augmentation (possibly monocular)
- location-dependent body-stabilized augmentation (on a projection cylinder/sphere surrounding the user)
- stereo head-tracked, position tracked, AR with full overlay registration



Mobile AR – Challenges

Mobile AR is difficult



- “Basic” wearable computing is already a technical challenge. Mobile AR adds a lot of extra complexity: orientation & long-range position tracking, possibly 3D graphics...
- Ruggedness required (“wear and tear!” ☹)
- Outdoor AR is a particular challenge (wide range of operating conditions, little control over environment).



Mobile AR - Challenges (2)



Limited Resources

- A wearable platform has limited computation power
- Size, weight, and power restrictions:
 - Military backpacks can weigh about 60 pounds (27 kg), military helmets 4-5 pounds (~2kg)
 - For a system to *appeal* to users, the weight has to be *drastically* lower and the ergonomics have to be right.
 - Batteries, batteries, batteries (esp. for 3D graphics)



Mobile AR - Hardware



Hardware requirements (tracking covered in next section):

- **Head-worn display**
 - Optical see-through vs. video feed-through, monocular vs binocular, stereo vs mono, resolution, field of view
 - Extra brightness for outdoors. Optical see-through: adjustable opacity
- **Computing Platform**
 - Computing power, 3D graphics capabilities, extensibility
 - Size, ergonomics, availability (off-the-shelf vs. build yourself), price
- **Complementary hand-held/palm-top/wrist displays**
 - For outdoors: readability in direct sunlight
- **(Other) input devices**
 - Mice, 3D pointing, microphones, cameras



Mobile AR - Hardware Head-Worn Displays



Optical see-through stereo displays



Sony Glasstron LDI-D100B (discontinued)



i-glasses 3D



Kaiser ProView series (here: PV40)



Mobile AR - Hardware
Head-Worn Displays



Minolta
Forgettable Display



MicroVision Nomad
retinal scanning display




MicroOptical EG 7



Mobile AR - Hardware
Head-Worn Displays



More resources (non-comprehensive)

VRNews article (January 2001):
<http://www.vrnews.com/issuearchive/vrn1001/vrn1001tech.html>

Daeyang Cy-Visor: http://www.personaldisplay.com/english/f_whatish.html
<http://www.cwonline.com/cyvisor.asp>

Olympus: <http://www.olympus-eye-trek.com>

Kaiser E-O: <http://www.keo.com/displayproducts.htm>

i-glasses: <http://www.i-glasses.com>

MicroVision: <http://www.mvis.com>

MicroOptical: <http://www.microopticalcorp.com>



Mobile AR - Hardware



Mixed Reality Systems Lab
Custom-designed hardware (used in TOWNWEAR' system):



- Bright optical see-through HMD with adjustable transmittance and embedded video camera
- Fiber optic gyroscope (TISS-5-40) and vision-based drift corrections for tracking head orientation

© Mixed Reality Systems Laboratory Inc. *'Towards Outdoor Wearable Navigator With Enhanced & Augmented Reality*



Mobile AR - Hardware Computing Platform



Current wearable/mobile solutions

Decision factors: compute power needed, form factor, OS, expansion ports, interface ports, memory, upgradeability, power consumption, support, price

- Buy a commercial *wearable*, custom made or kit (e.g. Xybernaut, Charmed,...)
- Choose and buy the components and assemble your own
- Use the (sub-)notebook of your choice, maybe modify it (e.g. take off screen)



Mobile AR - Hardware 3D Graphics Platform



Current 3D graphics solutions

Decision factors: graphics power needed, API support, availability of stereo drivers, power consumption, video memory, price

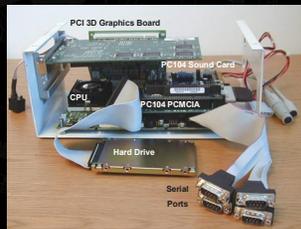
- Wait for a commercial 3D *wearable*...
- Notebooks/laptops with fast 3D accelerators
 - GeForce2 Go chip: Toshiba Satellite 5805-402, Dell Inspiron 8000
 - Others: S3 (Super) Savage/MX Mobile, ATI Rage M4 (mobile Radeon announced)
- Building mobile platform with PCI support:
 - nVidia GeForce2 MX 200/400, older FireGL boards, ...
- Building mobile platform with AGP support



Mobile AR - Hardware Computing Platform



Example self-built working solution with PCI-based 3D graphics



Columbia Touring Machine

Mobile AR - Tracking Position Tracking



Stay away from tether and do one or both of the following:

a) Equip environment with sensors, beacons, or visual fiducials

sensors example: UNC's *Sea of Cameras* ('94)
beacon examples: GPS or MIT's *Locust Swarm* ('97)
visual fiducials: ARToolKit

b) Rely on local sensors (carried/worn by the user)

E.g., fiducial-free vision-based approaches or dead-reckoning, based on orientation trackers, accelerometers, pedometers, odometers, etc.)



Mobile AR - Tracking Position Tracking



Indoor Solutions:

- Sparsely placed infrared beacons (MIT's *locust swarm*)
- Array of ultrasound sensors (AT&T Cambridge's *Bats*)
- Various local sensor (dead-reckoning) approaches
- Local sensor approaches combined with IR and other sensing
- Vision-based approaches (e.g. GVU's ('00) continuous path tracking using omnidirectional imagery)
- First attempts at using ARToolKit for long-range indoor tracking (U. of South Australia's *ARQuake*)



Mobile AR - Tracking Position Tracking



Outdoor Solutions:

- GPS, differential GPS, RTK differential GPS
- GPS combined with dead-reckoning
- Pseudolites (terrestrial GPS transceivers)
- Vision based techniques are being explored (by now no truly general or real-time solutions exist)



Mobile AR - Tracking Position Tracking



Solutions that can be used both in- and outdoors:

- Mobile-phone-based approaches:
 - simple cell identification (but cells vary considerably in size)
 - triangulation of time-of-flight information for the radio signals to three or more base stations.
- Terrestrial network of GPS transceivers (pseudolites)
- Local-sensor-based approaches (e.g., dead reckoning based on orientation trackers, accelerometers, pedometers, odometers, ...)
- Complementary hybrids



Mobile AR - Tracking Orientation Tracking



General Approaches:

- Magnetometers, Inclometers, Gyroscopes
- Fiber Optic Gyroscope (FOG TISS-5-40 mentioned before)
- Distortion compensation for magnetometers

The FOG's performance is promising, but hybrid approaches can combine multiple sensors to cover weaknesses



Mobile AR - Tracking Orientation Tracking



Hybrid Approaches:

- UNC optical tracker [Azuma94]
- Rockwell's silhouette matching
- InterSense hybrid trackers [Foxlin98]
- Inertial-optical hybrids (HIRL and USC)



Mobile AR - Tracking
Hybrid Orientation Tracking



CG & UI Laboratory
Columbia University

Outdoor motion-stabilized AR (Azuma '99,'00):

© HRL Laboratories

60 Hz update rate,
peak errors < $\sim 2^\circ$,
average errors < 1°



Mobile AR - Tracking
Hybrid Orientation Tracking



CG & UI Laboratory
Columbia University

Outdoor motion-stabilized AR (Azuma '99,'00):

Video
© HRL Laboratories



Environmental Modeling



CG & UI Laboratory
Columbia University

Unless we are attaching information to markers in the scene only,
the computer needs a model of the environment



- For annotating detailed infrastructure: need *geometrical* model
- Access to DB of environmental information



Environmental Modeling

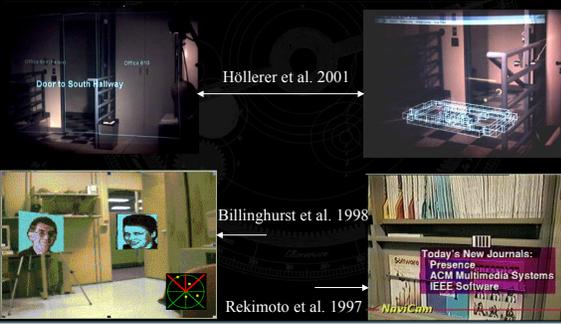


- Model urban infrastructure from 2D topographic maps and aerial photographs
- Modeling from laser range finder data
- Modeling from a combination of a set of photographs and geometrical constraints (Berkeley Façade, Canoma)



Mobile AR Systems (MARS)

Indoor AR



MARS

Indoor AR



Indoor mobile AR systems

- Sony CSL: NaviCam
- MIT: AR through wearable computing
- HITLab: Shared Space (many user studies)
- Uni Saarbrücken: Augmenting buildings with strong IR senders
- Georgia Tech, U. of South Australia: AR Gaming



Mobile AR Systems

Outdoor AR



Existing Outdoor Systems

- focus on tracking:
 - HRL, Rockwell, USC, Mixed Reality Systems Lab, ...
- focus on systems/UI:
 - Columbia University, University of South Australia, Naval Research Lab, Mixed Reality Systems Lab
 - Papers/Posters at ISAR and ISMR symposia



Mobile AR Systems

Outdoor AR



Columbia University MARS



- Touring Machine ('97), Situated Documentaries ('99)
- Indoor/Outdoor Collaboration ('99),
- Filtering ('00, with NRL), View Mgmt ('01)



Mobile AR Systems

Outdoor AR



University of South Australia system (Tinmith-4, ARQuake)



© University of South Australia

- Terrestrial Navigation ('98), VR/AR ('99)
- ARQuake ('00): Outdoor/Indoor game, vision-based tracking corrections (ARToolKit)



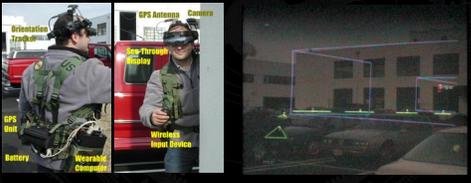
Mobile AR Systems

Outdoor AR



CG & UI Laboratory
Columbia University

NRL's Battlefield Augmented Reality System



Naval Research Laboratory

- BARS ('00), Information Filtering ('00)
- Focuses on stereo 3D Vector graphics (also supports polygonal 3D models)



Mobile AR Systems

Outdoor AR



CG & UI Laboratory
Columbia University

Mixed Reality Systems Lab



© Mixed Reality Systems Laboratory Inc.

- TOWNWEAR (Towards Outdoor Wearable Navigator With Enhanced & Augmented Reality)
- Head orientation tracking with fiber optic gyroscope and vision-based drift corrections



MARS

UI Case Study



CG & UI Laboratory
Columbia University

Columbia MARS Testbed

(1996 – today)



MARS
UI Case Study



CO & U Laboratory
Columbia University

Touring Machine: Assist mobile user in exploring unfamiliar environment



User interface hardware

- See-through head-worn display and tracker
- Hand-held pen computer/tablet

SIGGRAPH
2001
EXPLORING INTERACTION AND DIGITAL IMAGES

MARS
Columbia Touring Machine



CO & U Laboratory
Columbia University



Tablet computer with 2D/3D campus map

Map interface supports navigation and information queries

SIGGRAPH
2001
EXPLORING INTERACTION AND DIGITAL IMAGES

MARS
Mobile Journalist's Workstation



CO & U Laboratory
Columbia University

Mobile AR system as a journalistic tool

- for news producers ("one-person-broadcast-van")
- for news consumers:



Embed news stories and historic reports in actual event location
Example: Columbia student revolt/strike of 1968

(with John Pavlik, Columbia School of Journalism)

SIGGRAPH
2001
EXPLORING INTERACTION AND DIGITAL IMAGES

MARS

Situated Documentaries



CO & U Laboratory
Columbia University

Three main story threads:



Student revolt
of 1968



Bloomingdale
Asylum



Columbia
tunnel system



Situated Documentaries

Physical Hypermedia

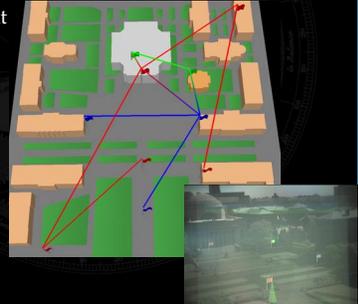


CO & U Laboratory
Columbia University

Virtual flags represent points of interest

Selecting Flags:

- Visual select
- Positional proximity
- Selection from list
- Following links



Situated Documentaries

Context Menus

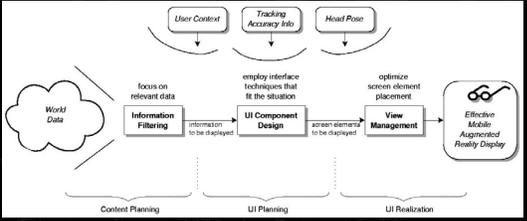


CO & U Laboratory
Columbia University

Screen-stabilized and world-stabilized elements




UI considerations

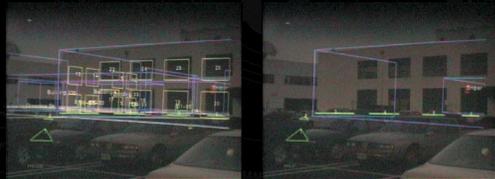
Context Planning UI Planning UI Realization



UI considerations



Information Filtering (Julier et al. '00)



- Remove clutter by goal- and distance based filtering
- User's task is route finding; Sniper and relevant buildings are displayed; objects, which are determined to be unnecessary, removed

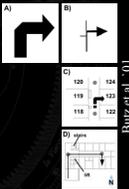


UI considerations



UI component design (UI planning):

- McIntyre and Coelho ('00) adapt the display to dynamic registration errors
- Butz et al. ('01) plan graphical way description schemata according to tracking quality
- The Columbia system reacts to changes in position tracking accuracy




Hollerer et al. '01

UI considerations



View Management:

- Labels are laid out dynamically to annotate campus model as seen by the observer.
- UI elements avoid overlapping the colleague's head and the campus model.



Acknowledgments



ONR Contracts N00014-99-1-0249,
N00014-99-1-0394, and
N00014-99-0683
NSF Grant IIS-00-82961
Gifts from Intel, Microsoft, Mitsubishi,
and IBM



**Developing Applications
with ARToolKit**

Hirokazu Kato
Hiroshima City University
kato@sys.im.hiroshima-cu.ac.jp
<http://www.sys.im.hiroshima-cu.ac.jp/people/kato/>



Outline

1. What is ARToolKit?
2. How does ARToolKit work?
3. Steps for developing an application
4. Camera calibration
5. ARToolKit-based interaction methods
6. Demonstrations



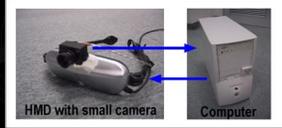
What is ARToolKit?

- **Library for vision-based AR applications**
 - Open Source, multi-platform
- **Overlays 3D virtual objects on real markers**
 - Uses single tracking marker
 - Determines camera pose information (6 DOF)
- **Includes utilities for marker-based interaction**
- **ARToolKit Website**
http://www.hitl.washington.edu/research/shared_space/



Hardware

- **Camera**
 - 320x240+
- **Computer**
 - Pentium 500Mhz+
 - 3D graphics video card
 - Video capture card
- **HMD (optional)**
 - Video see-through or Optical see-through
 - Binocular or Monocular



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Typical ARToolKit System

- Pentium III 800 Mhz - \$1200
- GeForce2 GTS Graphics - \$250
- Hauppauge WinTV capture card - \$50
- Marshall Board CCD Camera - \$250
- Sony Glastron PLM-A35 - \$400
- VGA to NTSC converter - \$100

Total Cost ~ \$2250

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Software

- **ARToolKit : version 2.40 or later**
 - libAR - tracking
 - libARVideo - video capturing
 - libARgsub - image drawing
- **OS: Linux, IRIX, Windows**
- **Language: C**

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

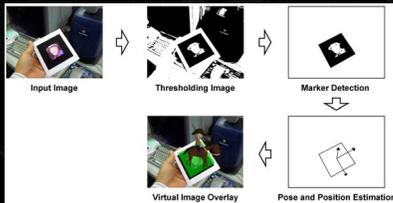
Software (cont.)

- **Additional basic libraries**
 - Video capture library (Video4Linux, VisionSDK)
 - OpenGL
 - GLUT
- **Other useful libraries**
 - Open VRML, Open Inventor, WTK, etc

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

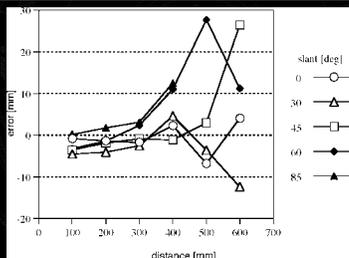
How does ARToolKit work?

- **Tracking overview**



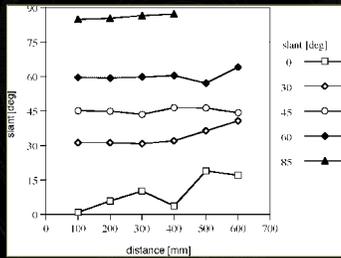
SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Tracking Error with Distance



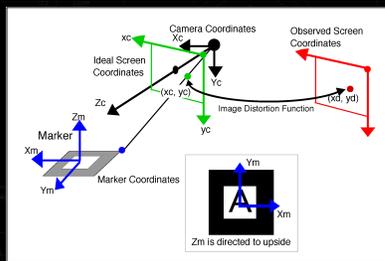
SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Tracking Error with Marker Angle



SIGGRAPH 2001
EXPLORING INTERACTION AND DIGITAL IMAGES

Coordinate Systems



SIGGRAPH 2001
EXPLORING INTERACTION AND DIGITAL IMAGES

Required Parameters

- **Camera parameters**
 - Transformation from camera coordinates to ideal screen coordinates
 - Image distortion function
=> Camera calibration utility program
- **Definition of marker coordinates**
 - Origin and Size
- **Pattern in the marker**
 - Pattern template

SIGGRAPH 2001
EXPLORING INTERACTION AND DIGITAL IMAGES

Camera calibration

- Using dot pattern and grid pattern

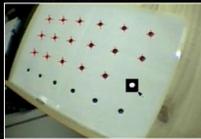


- 2 step method

- 1) Getting distortion parameters
- 2) Getting perspective geometric camera parameters



Camera calibration - step 1



Selecting dots with mouse



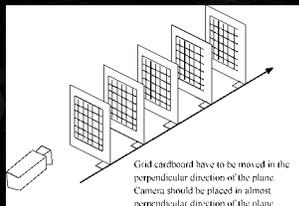
Getting distortion parameters by automatic line-fitting



Camera calibration - step 2



Manual line-fitting



Grid cardboard have to be moved in the perpendicular direction of the plane. Camera should be placed in almost perpendicular direction of the plane.



Steps for Developing a Simple AR Application

- Ex. 1: Simple video display
- Ex. 2: Detecting a marker
- Ex. 3: Using pattern
- Ex. 4: Getting a 3D information
- Ex. 5: Virtual object overlay
- Ex. 6: Using complex objects
- Tips

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Ex.1: Simple video display

- Program : sample1.c
- Key points
 - Loop structure
 - Video image handling
 - Camera parameter handling
 - Window setup
 - Mouse and keyboard handling

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Sample1.c - main function

```
main()  
{  
    init();  
  
    arVideoCapStart();  
    argMainLoop(mouseEvent, keyEvent, mainLoop);  
}
```

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Sample1.c – mailLoop function

```
if( (dataPtr = (ARUint8 *)
      arVideoGetImage()) == NULL ) {
    arUtilsSleep(2);
    return;
}

argDrawMode2D();
argDispImage( dataPtr, 0, 0 );
arVideoCapNext();
argSwapBuffers();
```

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Sample1.c – video initialization

```
/* open the video path */
if( arVideoOpen("") < 0 ) exit(0);

/* find the size of the window */
if( arVideoInqSize(&xsize, &ysize) < 0 )
    exit(0);
printf("Image size (x,y) = (%d,%d)\n",
       xsize, ysize);
```

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Sample1.c – camera parameters

```
/* set the initial camera parameters */
if( arParamLoad(CAMERA_PARAMETER_FILE, 1,
                &wparam) < 0 ) {
    printf("Camera parameter load error !!\n");
    exit(0);
}
arParamChangeSize( &wparam, xsize, ysize,
                  &cparam );
arInitCparam( &cparam );
```

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Ex. 2: Detecting a marker

- Program : sample2.c
- Key points
 - Threshold value
 - Important external variables
 - arDebug - keep thresholded image
 - arImage - pointer for thresholded image
 - arImageProcMode - use 50% image for image processing

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Sample2.c - marker detection

```
/* detect the markers in the video frame */  
if( arDetectMarker(dataPtr, thresh,  
    &marker_info, &marker_num) < 0 ) {  
    cleanup();  
    exit(0);  
}  
  
for( i = 0; i < marker_num; i++ ) {  
    argDrawSquare(marker_info[i].vertex,0,0);  
}
```

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

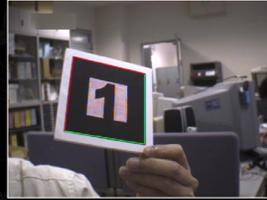
Ex. 3: Using pattern

- Program : sample3.c
- Key points
 - Pattern files loading
 - Structure of marker information
 - Region features
 - Pattern Id, direction
 - Certainty factor
 - Marker identification

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Making a pattern template

- Use of utility program:
mk_patt
- Show the pattern
- Put the corner of red line segments on the left-top vertex of the marker



SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Sample3.c – pattern files loading

```
/* load pattern file */
if((patt_id1=arLoadPatt(PATTERN_FILE1)) < 0){
    printf("Pattern file load error !!\n");
    exit(0);
}
if((patt_id2=arLoadPatt(PATTERN_FILE2)) < 0){
    printf("Pattern file load error !!\n");
    exit(0);
}
```

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Sample3.c – marker_info structure

```
typedef struct {
    int    area;
    int    id;
    int    dir;
    double cf;
    double pos[2];
    double line[4][3];
    double vertex[4][2];
} ARMarkerInfo;
```

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Ex. 4: Getting a 3D information

- Program : sample4.c
- Key points
 - Definition of a real marker
 - Transformation matrix
 - Rotation component
 - Translation component

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Sample4.c - getting a transformation matrix

```
double marker_center[2] = {0.0, 0.0};  
double marker_width = 40.0;  
double marker_trans[3][4];  
  
arGetTransMat(&marker_info[i], marker_center,  
marker_width, marker_trans);
```

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Ex. 5: Virtual object overlay

- Program : sample5.c
- Key points
 - OpenGL parameter setting
 - Setup of projection matrix
 - Setup of modelview matrix

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Sample5.c - OpenGL setup for 3D objects

```
/* setup the projection matrix */  
argDrawMode3D();  
argDraw3dCamera( 0, 0 );  
  
/* load the camera transformation matrix */  
argConvGlpara(trans, gl_para);  
glMatrixMode(GL_MODELVIEW);  
glLoadMatrixd( gl_para );
```

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Ex. 6: Using complex objects

- VRML
 - Open VRML - <http://www.openvrm.org/>
- Others
 - Open Inventor
 - WTK

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

ARToolKit-based Interaction Methods

- What kind of information can we use?
 - Spatial relationship
 - Movement
- 
 - Marker ⇔ User's Head
 - Marker ⇔ Marker
- Using the transformation matrix

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Relationship between user's head and marker

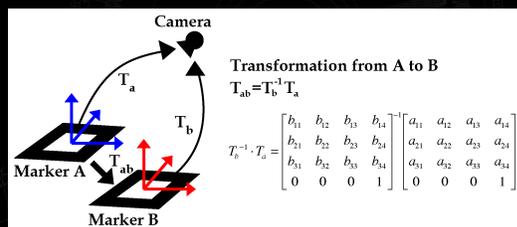
- `double trans[3][4];`
 - `arGetTransMat();`
 - Transformation matrix from marker coordinates to camera coordinates

$$\begin{bmatrix} R_{11} & R_{12} & R_{13} & T_x \\ R_{21} & R_{22} & R_{23} & T_y \\ R_{31} & R_{32} & R_{33} & T_z \end{bmatrix}$$

R: Rotation - Marker orientation in camera coordinates
T: Translation - Marker position in camera coordinates



Relationship between two markers

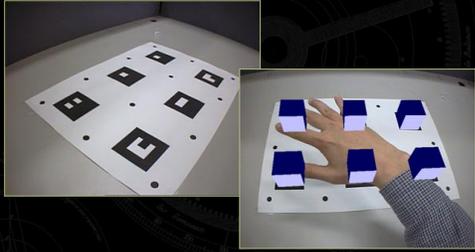


More Complex Demonstrations

- **ExView**
 - Display the relationship between camera and marker
- **Multi-marker Tracking**
 - Using multiple markers for robust tracking
- **VOMAR**
 - Interaction with virtual objects and real objects

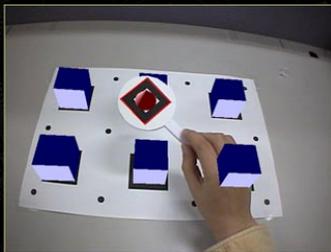


Multi-Marker Tracking



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Real Object Occlusion



SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Tips

- **Image size, speed, accuracy**
 - Input image size - `arVideoOpen()`;
 - For image processing
 - Re-sample - `arImageProcMode`;
 - For display
 - Zoom parameter - `argInit()`;
 - Texture mapping v.s. `glDrawPixels()`;
external variable: `argDrawMode`

SIGGRAPH
2001
EXPLORE INTERACTION
AND DIGITAL IMAGES

Tips 2

- **Interlaced image is not good for tracking.**
 - Full size input and half size image processing
 - Half size input and same size image processing
- **Frame display (640x480)
v.s. field display (640x240)**
 - Image quality
 - Speed
 - External variable: argTexmapMode

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Future Directions

- **ARToolKit in the future:**
 - Firewire tracking
 - Hybrid tracking
 - Vision + inertial/magnetic
 - Natural feature recognition

SIGGRAPH
2001
EXPLORING INTERACTION
AND DIGITAL IMAGES

Sample source codes for ARToolkit

```
/* ----- sample1.c ----- */
#include <windows.h>
#include <stdio.h>
#include <GL/glut.h>
#include <AR/gsub.h>
#include <AR/video.h>
#include <AR/param.h>
#include <AR/ar.h>

#define CAMERA_PARAMETER_FILE "Data/camera_para.dat"

static void init(void);
static void cleanup(void);
static void keyEvent( unsigned char key, int x, int y);
static void mouseEvent(int button, int state, int x, int y);
static void mainLoop(void);

main()
{
    init();
    arVideoCapStart();
    argMainLoop( mouseEvent, keyEvent, mainLoop );
}

static void keyEvent( unsigned char key, int x, int y)
{
    /* quit if the ESC key is pressed */
    if( key == 0x1b ) {
        cleanup();
        exit(0);
    }
}

/* mouse event handling function */
static void mouseEvent(int button, int state, int x, int y)
{
    if( button == GLUT_LEFT_BUTTON && state == GLUT_UP ) {
        cleanup();
        exit(0);
    }
}

/* main loop */
static void mainLoop(void)
{
    ARUint8 *dataPtr;

    /* grab a vide frame */
    if( (dataPtr = (ARUint8 *)arVideoGetImage()) == NULL ) {
        arUtilSleep(2);
        return;
    }
    argDrawMode2D();
    argDispImage( dataPtr, 0, 0 );
    arVideoCapNext();
    argSwapBuffers();
}

static void init( void )
{
    ARParam wparam, cparam;
    int xsize, ysize;

    /* open the video path */
    if( arVideoOpen("") < 0 ) exit(0);
    /* find the size of the window */
    if( arVideoInqSize(&xsize, &ysize) < 0 ) exit(0);
    printf("Image size (x,y) = (%d,%d)\n", xsize, ysize);

    /* set the initial camera parameters */
    if( arParamLoad(CAMERA_PARAMETER_FILE, 1, &wparam) < 0 ) {
        printf("Camera parameter load error !!\n");
        exit(0);
    }
    arParamChangeSize( &wparam, xsize, ysize, &cparam );
    arInitCparam( &cparam );

    /* open the graphics window */
    argInit( &cparam, 1.0, 0, 0, 0, 0 );
}

/* cleanup function called when program exits */
static void cleanup(void)
{
    arVideoCapStop();
    arVideoClose();
    argCleanup();
}
}
```

```

/* ----- sample2.c ----- */
#include <_WIN32>
#include <windows.h>
#include <stdio.h>
#include <GL/glut.h>
#include <AR/gsub.h>
#include <AR/video.h>
#include <AR/param.h>
#include <AR/ar.h>

#define CAMERA_PARAMETER_FILE "Data/camera_para.dat"

int thresh = 100;

static void init(void);
static void cleanup(void);
static void keyEvent( unsigned char key, int x, int y );
static void mainLoop(void);

main()
{
    init();
    arVideoCapStart();
    argMainLoop( NULL, keyEvent, mainLoop );
}

static void keyEvent( unsigned char key, int x, int y )
{
    /* quit if the ESC key is pressed */
    if( key == 0x1b ) {
        cleanup();
        exit(0);
    }

    if( key == 'd' ) arDebug = 1 - arDebug;

    if( key == 'r' ) {
        switch( arImageProcMode ) {
            case AR_IMAGE_PROC_IN_FULL:
                arImageProcMode = AR_IMAGE_PROC_IN_HALF;
                break;
            case AR_IMAGE_PROC_IN_HALF:
                arImageProcMode = AR_IMAGE_PROC_IN_FULL;
                break;
        }
    }

    if( key == 't' ) {
        printf("Enter new threshold value (now = %d): ", thresh);
        scanf("%d",&thresh); while( getchar()!='\n' );
        printf("\n");
    }
}

}

/* main loop */
static void mainLoop(void)
{
    ARUint8 *dataPtr;
    ARMarkerInfo *marker_info;
    int marker_num;
    int i;

    /* grab a vide frame */
    if( (dataPtr = (ARUint8 *)arVideoGetImage()) == NULL ) {
        arUtilsSleep(2);
        return;
    }

    /* detect the markers in the video frame */
    if( arDetectMarker(dataPtr, thresh, &marker_info, &marker_num)
        < 0 ) {
        cleanup();
        exit(0);
    }

    arDrawMode2D();
    if( arDebug == 0 ) {
        arDispImage( dataPtr, 0, 0 );
    }
    else {
        if( arImageProcMode == AR_IMAGE_PROC_IN_HALF )
            arDispHalfImage( arImage, 0, 0 );
        else
            arDispImage( arImage, 0, 0 );
    }
    arVideoCapNext();

    glColor3f( 1.0, 0.0, 0.0 );
    glLineWidth( 3.0 );
    for( i = 0; i < marker_num; i++ ) {
        arDrawSquare( marker_info[i].vertex, 0, 0 );
    }
    glLineWidth( 1.0 );
    argSwapBuffers();
}

static void init( void )
{
    ARParam wparam, cparam;
    int xsize, ysize;

    /* open the video path */
    if( arVideoOpen("") < 0 ) exit(0);
}

```

```

/* find the size of the window */
if( arVideoInqSize(&xsize, &ysize) < 0 ) exit(0);
printf("Image size (x,y) = (%d,%d)\n", xsize, ysize);

/* set the initial camera parameters */
if( arParamLoad(CAMERA_PARAMETER_FILE, 1, &wparam) < 0 ) {
    printf("Camera parameter load error !!\n");
    exit(0);
}
arParamChangeSize( &wparam, xsize, ysize, &cparam );
arInitCparam( &cparam );

/* open the graphics window */
argInit( &cparam, 1.0, 0, 0, 0, 0 );

}

/* cleanup function called when program exits */
static void cleanup(void)
{
    arVideoCapStop();
    arVideoClose();
    argCleanup();
}

}

/* ----- sample3.c ----- */
#include <stdio.h>
#include <stdlib.h>
#include <GL/glut.h>
#include <AR/gsub.h>
#include <AR/video.h>
#include <AR/param.h>
#include <AR/ar.h>

#define CAMERA_PARAMETER_FILE "Data/camera_para.dat"
#define PATTERN_FILE1 "Data/samplePatt1"
#define PATTERN_FILE2 "Data/samplePatt2"

int thresh = 100;
int patt_id1;
int patt_id2;

static void init(void);
static void cleanup(void);
static void keyEvent( unsigned char key, int x, int y);
static void mainLoop(void);

main()
{
    init();
    arVideoCapStart();
    argMainLoop( NULL, keyEvent, mainLoop );
}

static void keyEvent( unsigned char key, int x, int y)
{
    /* quit if the ESC key is pressed */
    if( key == 0x1b ) {
        cleanup();
        exit(0);
    }

    if( key == 't' ) {
        printf("Enter new threshold value (now = %d): ", thresh);
        scanf("%d",&thresh); while( getchar() != '\n' );
        printf("\n");
    }
}

/* main loop */
static void mainLoop(void)
{
    ARUint8 *dataPtr;
    ARMarkerInfo *marker_info;
    int marker_num;
}

```

```

int      i;

/* grab a vide frame */
if( (dataPtr = (ARUint8 *)arVideoGetImage()) == NULL ) {
    arUtilSleep(2);
    return;
}

/* detect the markers in the video frame */
if( arDetectMarker(dataPtr, thresh, &marker_info, &marker_num)
< 0 ) {
    cleanup();
    exit(0);
}

argDrawMode2D();
argDispImage( dataPtr, 0, 0 );
arVideoCapNext();

printf("Marker_num = %d\n", marker_num);
gLineWidth( 3.0 );
for( i = 0; i < marker_num; i++ ) {
    printf("%2d: area(%d), pos(%5.1f,%5.1f), id(%d), cf(%f)\n",
        i, marker_info[i].area,
        marker_info[i].pos[0], marker_info[i].pos[1],
        marker_info[i].id,
        marker_info[i].cf);
}

if( marker_info[i].id < 0 ) glColor3f( 1.0, 0.0, 0.0 );
else if( marker_info[i].id == patt_id1 ) glColor3f( 0.0,
1.0, 0.0 );
else if( marker_info[i].id == patt_id2 ) glColor3f( 0.0,
0.0, 1.0 );
    argDrawSquare( marker_info[i].vertex, 0, 0 );
}
gLineWidth( 1.0 );
argSwapBuffers();
}

static void init( void )
{
    ARParam  wparam, cparam;
    int      xsize, ysize;

/* open the video path */
if( arVideoOpen("") < 0 ) exit(0);
/* find the size of the window */
if( arVideoInqSize(&xsize, &ysize) < 0 ) exit(0);
printf("Image size (x,y) = (%d,%d)\n", xsize, ysize);

/* set the initial camera parameters */
if( arParamLoad(CAMERA_PARAMETER_FILE, 1, &wparam) < 0 ) {
    printf("Camera parameter load error !\n");
    exit(0);
}
arParamChangeSize( &wparam, xsize, ysize, &cparam );
arInitCparam( &cparam );

/* load pattern file */
if( (patt_id1 = arLoadPatt(PATTERN_FILE1)) < 0 ) {
    printf("Pattern file load error !\n");
    exit(0);
}
if( (patt_id2 = arLoadPatt(PATTERN_FILE2)) < 0 ) {
    printf("Pattern file load error !\n");
    exit(0);
}

/* open the graphics window */
argInit( &cparam, 1.0, 0, 0, 0, 0 );
}

/* cleanup function called when program exits */
static void cleanup(void)
{
    arVideoCapStop();
    arVideoClose();
    argCleanup();
}
}

```

```

/* ----- sample4.c ----- */
#include <_WIN32>
#include <windows.h>
#include <stdio.h>
#include <GL/glut.h>
#include <AR/gsub.h>
#include <AR/video.h>
#include <AR/param.h>
#include <AR/ar.h>

#define CAMERA_PARAMETER_FILE "Data/camera_para.dat"
#define PATTERN_FILE "Data/samplePatt1"

int thresh = 100;
int patt_id1;
double marker_center[2] = {0.0, 0.0};
double marker_width = 40.0;
double marker_trans[3][4];

static void init(void);
static void cleanup(void);
static void keyEvent( unsigned char key, int x, int y);
static void mainLoop(void);

main()
{
    init();
    arVideoCapStart();
    argMainLoop( NULL, keyEvent, mainLoop );
}

static void keyEvent( unsigned char key, int x, int y)
{
    /* quit if the ESC key is pressed */
    if( key == 0x1b ) {
        cleanup();
        exit(0);
    }

    if( key == 't' ) {
        printf("Enter new threshold value (now = %d): ", thresh);
        scanf("%d",&thresh); while( getchar()!='\n' );
        printf("\n");
    }

    /* main loop */
    static void mainLoop(void)
    {
        ARUint8 *dataPtr;
        ARMarkerInfo *marker_info;

        int marker_num;
        int i;

        /* grab a vide frame */
        if( (dataPtr = (ARUint8 *)arVideoGetImage()) == NULL ) {
            arUtilsSleep(2);
            return;
        }

        /* detect the markers in the video frame */
        if( arDetectMarker(dataPtr, thresh, &marker_info, &marker_num)
            < 0 ) {
            cleanup();
            exit(0);
        }

        argDrawMode2D();
        argDispImage( dataPtr, 0, 0 );
        arVideoCapNext();

        printf("Marker_num = %d\n", marker_num);
        glLineWidth( 3.0 );
        for( i = 0; i < marker_num; i++ ) {
            printf("%2d: area(%d), pos(%5.1f,%5.1f), id(%d), cf(%f)\n",
                i, marker_info[i].area,
                marker_info[i].pos[0], marker_info[i].pos[1],
                marker_info[i].id, marker_info[i].cf);
        }

        if( marker_info[i].id != patt_id1 ) continue;
        glColor3f( 1.0, 0.0, 0.0 );
        argDrawSquare( marker_info[i].vertex, 0, 0 );
        argGetTransMat(&marker_info[i], marker_center, marker_width,
            marker_trans);

        printf("%5.1f %5.1f %5.1f\n", marker_trans[0][3],
            marker_trans[1][3], marker_trans[2][3] );
    }

    glLineWidth( 1.0 );
    argSwapBuffers();
}

static void init( void )
{
    ARParam wparam, cparam;
    int xsize, ysize;

    /* open the video path */
    if( arVideoOpen("") < 0 ) exit(0);
    /* find the size of the window */
    if( arVideoInqSize(&xsize, &ysize) < 0 ) exit(0);
    printf("Image size (x,y) = (%d,%d)\n", xsize, ysize);
}

```

```

/* set the initial camera parameters */
if( arParamLoad(CAMERA_PARAMETER_FILE, 1, &wparam) < 0 ) {
    printf("Camera parameter load error !!\n");
    exit(0);
}
arParamChangeSize( &wparam, xsize, ysize, &cparam );
arInitCparam( &cparam );

/* load pattern file */
if( ( patt_id1 = arLoadPatt(PATTERN_FILE1)) < 0 ) {
    printf("Pattern file load error !!\n");
    exit(0);
}

/* open the graphics window */
arginit( &cparam, 1.0, 0, 0, 0, 0 );

/* cleanup function called when program exits */
static void cleanup(void)
{
    arVideoCapStop();
    arVideoClose();
    argCleanup();
}

}

/* ----- sample5.c ----- */
#include <stdio.h>
#include <stdlib.h>
#include <GL/glut.h>
#include <AR/gsub.h>
#include <AR/video.h>
#include <AR/param.h>
#include <AR/ar.h>

#define CAMERA_PARAMETER_FILE "Data/camera_para.dat"
#define PATTERN_FILE1 "Data/samplePatt1"

int thresh = 100;
int patt_id1;
double marker_center[2] = { 0.0, 0.0 };
double marker_width = 40.0;
double marker_trans[3][4];

static void init(void);
static void cleanup(void);
static void keyEvent( unsigned char key, int x, int y);
static void mainLoop(void);
static void drawObject( double trans[3][4] );
static void initLights( void );

main()
{
    init();
    arVideoCapStart();
    argMainLoop( NULL, keyEvent, mainLoop );
}

static void keyEvent( unsigned char key, int x, int y)
{
    /* quit if the ESC key is pressed */
    if( key == 0x1b ) {
        cleanup();
        exit(0);
    }

    if( key == 't' ) {
        printf("Enter new threshold value (now = %d): ", thresh);
        scanf("%d", &thresh); while( getchar() != '\n' );
        printf("\n");
    }
}

```

```

/* main loop */
static void mainLoop(void)
{
    ARUint8      *dataPtr;
    ARMarkerInfo *marker_info;
    int          marker_num;
    int          i;

    /* grab a vide frame */
    if( (dataPtr = (ARUint8 *)arVideoGetImage()) == NULL ) {
        arUtilsSleep(2);
        return;
    }

    /* detect the markers in the video frame */
    if( arDetectMarker(dataPtr, thresh, &marker_info, &marker_num)
        < 0 ) {
        cleanup();
        exit(0);
    }

    argDrawMode2D();
    argDispImage( dataPtr, 0, 0 );
    arVideoCapNext();

    for( i = 0; i < marker_num; i++ ) {
        if( marker_info[i].id != patt_id1 ) continue;

        arGetTransMat(&marker_info[i], marker_center, marker_width,
                    marker_trans);
        drawObject( marker_trans );
    }

    argSwapBuffers();

    static void init( void )
    {
        ARParam wparam, cparam;
        int      xsize, ysize;

        /* open the video path */
        if( arVideoOpen("") < 0 ) exit(0);
        /* find the size of the window */
        if( arVideoInqSize(&xsize, &ysize) < 0 ) exit(0);
        printf("Image size (x,y) = (%d,%d)\n", xsize, ysize);

        /* set the initial camera parameters */
        if( arParamLoad(CAMERA_PARAMETER_FILE, 1, &wparam) < 0 ) {
            printf("Camera parameter load error !!\n");
            exit(0);
        }
        arParamChangeSize( &wparam, xsize, ysize, &cparam );

        arInitCparam( &cparam );

        /* load pattern file */
        if( (patt_id1 = arLoadPatt(PATTERN_FILE1)) < 0 ) {
            printf("Pattern file load error !!\n");
            exit(0);
        }

        /* open the graphics window */
        argInit( &cparam, 1.0, 0, 0, 0, 0 );

        /* cleanup function called when program exits */
        static void cleanup(void)
        {
            arVideoCapStop();
            arVideoClose();
            argCleanup();
        }

        /* material properties */
        GLfloat mat_flash[] = { 1.0, 0.0, 0.0, 1.0 };
        GLfloat mat_flash_shiny[] = { 25.0 };
        GLfloat mat_ambient[] = { 1.0, 0.0, 0.0, 1.0 };

        static void drawObject( double trans[3][4] )
        {
            double gl_para[16];

            /* setup the projection matrix */
            argDrawMode3D();
            argDraw3dCamera( 0, 0 );

            /* load the camera transformation matrix */
            argConvGpara(trans, gl_para);
            glMatrixMode(GL_MODELVIEW);
            glLoadMatrixd( gl_para );

            glClearDepth( 1.0 );
            glClear(GL_DEPTH_BUFFER_BIT);
            glDepthFunc(GL_EQUAL);
            initLights();

            glEnable(GL_DEPTH_TEST);
            glEnable(GL_LIGHTING);
            glEnable(GL_LIGHT0);
            glMaterialfv(GL_FRONT, GL_SPECULAR, mat_flash);
            glMaterialfv(GL_FRONT, GL_SHININESS, mat_flash_shiny);
            glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient);
            glutSolidCone(20.0, 40.0, 20, 24);
            glDisable(GL_LIGHT0);
            glDisable(GL_LIGHTING);
            glDisable(GL_DEPTH_TEST);
        }
    }
}

```

```
}
/* initialize the lights in the scene */
static void initLights( void )
{
    GLfloat light_position[] = {0.0, -200.0, 0.0, 0.0};
    GLfloat ambi[] = {0.1, 0.1, 0.1, 0.1};
    GLfloat lightZeroColor[] = {0.9, 0.9, 0.9, 0.1};

    glLightfv(GL_LIGHT0, GL_POSITION, light_position);
    glLightfv(GL_LIGHT0, GL_AMBIENT, ambi);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, lightZeroColor);
}
}
```

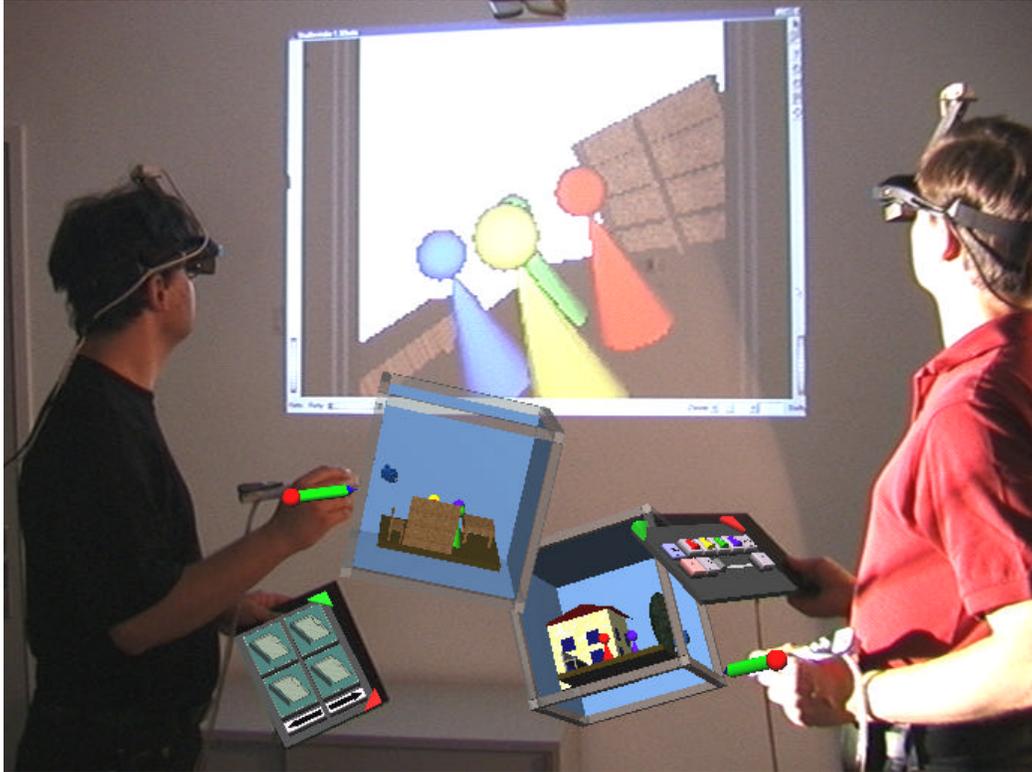


Figure 1.0: Two users collaborating on movie storyboard design. This heterogeneous setup combines see-through head-mounted displays with a large projection screen for additional information. The image is generated using video augmented reality in real-time using a camera and video workstation.

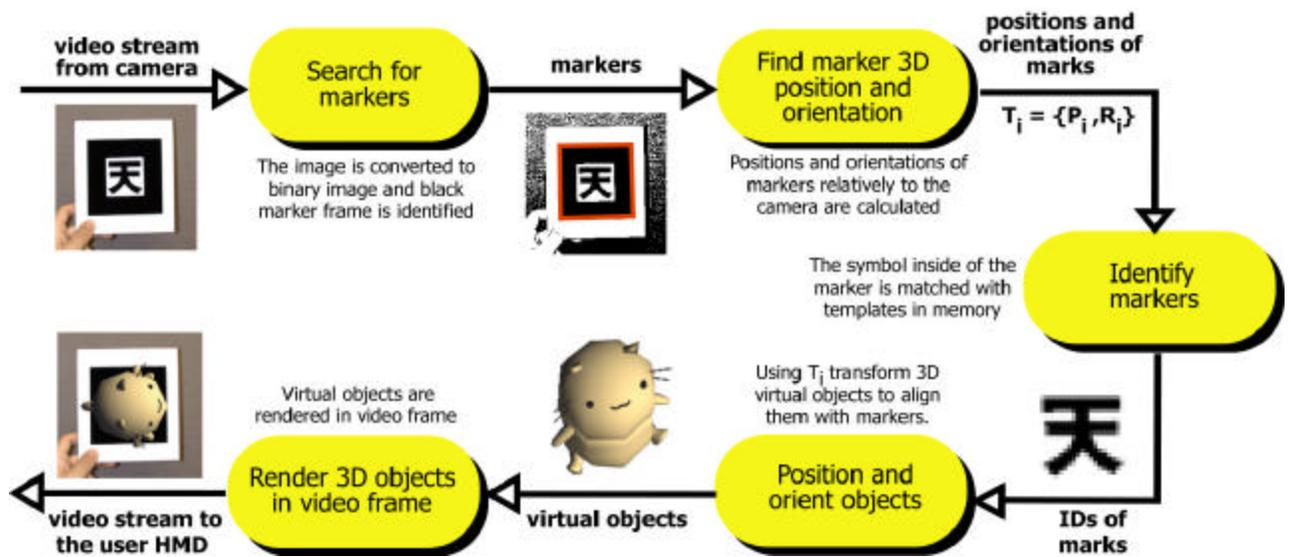


Figure 2.0: The ARToolKit computer-vision based tracking process.

Augmented Reality Bibliography

Amselen, D. A Window on Shared Virtual Environments. *Presence*, 1995, Vol. 4(2), pp. 130-145.

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.

ARGOS Virtual Pointer Camera Calibration Procedure. WWW page = http://vered.rose.utoronto.ca/people/david_dir/POINTER/Calibration.html

Azuma, R. (1993). Tracking Requirements for Augmented Reality. *Communications of the ACM*, 36(7), 50-51.

Azuma, R. and Bishop, G. (1994). Improving Static and Dynamic Registration in an Optical See-Through HMD. In *Proceedings of SIGGRAPH '94*, (pp. 197-204): ACM SIGGRAPH.

Azuma, R., Lee, J. W., Jiang, B., Park, J., You, S., and Neumann, U. (1999). Tracking in unprepared environments for augmented reality systems. *Computers and Graphics*, 23(6):787-793.

Azuma, Ronald T. A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments* 6, 4 (August 1997), 355 - 385. Earlier version appeared in Course Notes #9: Developing Advanced Virtual Reality Applications, *ACM SIGGRAPH '95* (Los Angeles, CA, 6-11 August 1995), 20-1 to 20-38.

Azuma, Ronald T. Augmented Reality: Approaches and Technical Challenges. In *Fundamentals of Wearable Computers and Augmented Reality*, Woodrow Barfield and Thomas Caudell, editors. Lawrence Erlbaum Associates, 2001, ISBN 0-8058-2901-6. Chapter 2, pp. 27-63.

Azuma, Ronald T. The Challenge of Making Augmented Reality Work Outdoors. In *Mixed Reality: Merging Real and Virtual Worlds*, Yuichi Ohta and Hideyuki Tamura, editors. Springer-Verlag, 1999, ISBN 3-540-65623-5. Chapter 21, pp. 379-390. Associated with invited presentation at the First International Symposium on Mixed Reality (ISMR '99) (Yokohama, Japan, 9-11 March 1999).

Azuma, Ronald T., Bruce R. Hoff, Howard E. Neely III, Ronald Sarfaty, Michael J. Daily, Gary Bishop, Vern Chi, Greg Welch, Ulrich Neumann, Suya You, Rich Nichols, and Jim Cannon. Making Augmented Reality Work Outdoors Requires Hybrid Tracking. *Proceedings of the First International Workshop on Augmented Reality*, (San Francisco, CA, 1 November 1998), 219-224.

Azuma, Ronald, and Gary Bishop. A Frequency-Domain Analysis of Head-Motion Prediction. *Proceedings of SIGGRAPH '95* (Los Angeles, CA, 6-11 August 1995). In Computer Graphics, Annual Conference Series, 1995, 401-408.

Azuma, Ronald, Bruce Hoff, Howard Neely III, Ron Sarfaty. A Motion-Stabilized Outdoor Augmented Reality System. *Proceedings of IEEE VR '99* (Houston, TX, 13-17 March 1999), 252-259.

Azuma, Ronald, Jong Weon Lee, Bolan Jiang, Jun Park, Suya You, and Ulrich Neumann. Tracking in unprepared environments for augmented reality systems. *Computers & Graphics* 23, 6 (December 1999), 787-793.

Azuma, Ronald. *Predictive Tracking for Augmented Reality*. Ph.D. Dissertation, University of North Carolina at Chapel Hill. Computer Science technical report TR#95-007, February 1995.

Bailiot, E. Gagas, T. Höllerer, S. Julier and S. Feiner, Y. (2000). Wearable 3D Graphics for Augmented Reality: A Case Study of Two Experimental Backpack Computers. *NRL Technical Report*.

Bajura, M. and Neumann, U. (1995). Dynamic Registration Correction in Augmented-Reality Systems. In *Proceedings of Virtual Reality Annual International Symposium, VRAIS '95*, (pp. 189-197): IEEE Computer Society Press.

Bajura, M., Fuchs, H., Ohbuchi, R. Merging Virtual Objects with the Real World: Seeing Ultrasound Imagery Within the Patient. In *Proceedings of SIGGRAPH '92*, 1992, New York: ACM Press, pp. 203-210.

Bajura, Mike. *Camera Calibration for Video See-Through Head-Mounted Display*. UNC Chapel Hill Department of Computer Science technical report TR93-048 (July 7, 1993), 6 pages.

Balcisoy, S. and Thalmann, D. (1997). Interaction between real and virtual humans in augmented reality. In *Proceedings of Computer Animation '97*, (pp. 31-38): IEEE Computer Society Press.

Bauer, M., Heiber, T., Kortuem, G., Segall, Z. A Collaborative Wearable System with Remote Sensing. In *Proceedings of the 2nd International Symposium on Wearable Computers*, October 1998, Pittsburgh, PA, pp. 10-17.

Bauer, M., Kortuem, G., Segall, Z. "Where Are You Pointing At?" A Study of Remote Collaboration in a Wearable Videoconference System. In *Proceedings of the 3rd International Symposium on Wearable Computers*, October 1999, San Francisco, CA, pp. 151-158.

Beadle, H., Harper, B., G. Maguire Jr., and Judge, J. (1997). Location aware mobile computing. In *Proc. ICT '97 (IEEE/IEE Int. Conf. on Telecomm.)*, Melbourne, Australia.

Behringer, R. (1999). Registration for outdoor augmented reality applications using computer vision techniques and hybrid sensors. In *Proc. IEEE Virtual Reality '99*, pages 244-251.

Berger, M. O. (1997). Resolving occlusion in augmented reality: a contour based approach without 3D reconstruction. In *Proceedings of 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, (pp. 91-96): IEEE Computer Society Press.

Billinghurst, M., Baldis, S., Miller, E., Weghorst, S. (1997). Shared Space: Collaborative Information Spaces. *Proceedings of the 7th International Conference on Human-Computer Interaction (HCI '97)*, August 24-29, 1997, San Francisco, USA.

Billinghurst, M., Bowskill, J., Dyer, N., Morphett, J. (1998). Evaluation of Spatial Interfaces for Wearable Computers. In *Proceedings of the Virtual Reality Annual International Symposium 1998 (VRAIS '98)*, March 14-18, 1998, Atlanta, Georgia, USA.

Billinghurst, M., Bowskill, J., Dyer, N., Morphett, J. (1998) Spatial Information Displays on a Wearable Computer. In *Computer Graphics and Applications*, IEEE Computer Society, November/December, 1998.

Billinghurst, M., Bowskill, J., Jessop, M. Morphett, J. (1998) A Wearable Spatial Conferencing Space. In *Proceedings of the Second International Symposium on Wearable Computing (ISWC '98)*, October 19th-20th, Pittsburgh, USA.

Billinghurst, M., Bowskill, J., Morphett, J. (1998) WearCom: A Wearable Communications Space. *British Telecom Journal*, October 1998.

Billinghurst, M., Kato, H., (1999) Collaborative Mixed Reality. In *Proceedings of the International Symposium on Mixed Reality (ISMR 99)*. March 9th-11th, Yokohama, Japan.

Billinghurst, M., Kato, H., (1999) Real World Teleconferencing. In *Proceedings of the conference on Human Factors in Computing Systems (CHI 99)*. May 15th-20th, Pittsburgh, USA.

Billinghurst, 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

Billinghurst, M., Poupyrev, I., Kato, H., May, R. Mixing Realities in Shared Space: An Augmented Reality Interface for Collaborative Computing. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME2000)*, July 30th - August 2, New York.

Billinghurst, M., Weghorst, S., Furness, T. (1998) Shared Space: An Augmented Reality Approach for Computer Supported Cooperative Work. *Virtual Reality : Research, Development and Application*, 1998.

Billigurst, M., Kato, H. (2000) Out and About: Real World Teleconferencing. *British Telecom Technical Journal (BTTJ)*, Millenium Edition, Jan 2000.

Bimber O., L. M. Encarnação, D. Schmalstieg: Enabling Back-Projection Support Systems to Support Augmented Reality. In *Proceedings of IEEE Virtual Reality 2000*, New Brunswick, New Jersey, March 18-22, 2000.

Borenstein, J., Everett, H., and Feng, L. (1996). *Navigating Mobile Robots: Systems and Techniques*. A K Peters Press, Natick, MA.

Bruce Hoff, Ronald Azuma. Autocalibration of an Electronic Compass in an Outdoor Augmented Reality System. *Proceedings of International Symposium on Augmented Reality 2000 (ISAR 2000)*, (Munich, Germany, 5-6 October 2000), 159-164.

Burbidge, Dick, and Paul M. Murray. Hardware Improvements to the Helmet-Mounted Projector on the Visual Display Research Tool (VDRT) at the Naval Training Systems Center. *SPIE Proceedings Vol. 1116 Head-Mounted Displays* (1989), 52-59.

Butz, A., Baus, J., and Krüger, A. (2000). Augmenting buildings with infrared information. In *Proceedings of the International Symposium on Augmented Reality (ISAR 2000)*, pages 93-96. IEEE Computer Society Press.

Butz, A., Beshers, C., and Feiner, S. (1998). Of vampire mirrors and privacy lamps: Privacy management in multi-user augmented environments. In *Proc. UIST'98*, pages 171-172. ACM SIGGRAPH.

Butz, A., Höllerer, T., Feiner, S., MacIntyre, B., Beshers, C. Enveloping Users and Computers in a Collaborative 3D Augmented Reality. In *Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality '99 (IWAR '99)*, October 20-21, San Francisco, CA, 1999, pp. 35-44.

Caudell, T. P. (1994). Introduction to augmented and virtual reality. In *Proceedings of Telem manipulator and Telepresence Technologies*, (pp. 272-281): SPIE.

Caudell, T.P., and Mizell, D.W. Augmented Reality: an application of heads-up display technology to manual manufacturing processes. In *Proceedings of the Twenty-Fifth Hawaii International Conference on Systems Science*, Kauai, Hawaii, 7th-10th Jan. 1992, Vol. 2, pp. 659-669.

Cavallaro, Rick. The FoxTrax Hockey Puck Tracking System. *IEEE Computer Graphics and Applications*, 17, 2 (March - April 1997), 6-12.

Cheverst, N. Davies, K. Mitchell and G. S. Blair, K. (2000). Developing a Context-aware Electronic Tourist Guide: Some Issues and Experiences. In *Proceedings of CHI 2000*, Netherlands.

Clarkson, B., Mase, K., and Pentland, A. (2000). Recognizing user context via wearable sensors. In *Proc. ISWC '00 (Fourth Int. Symp. on Wearable Computers)*, pages 69-75, Atlanta, GA.

Cohen, P., Johnston, M., McGee, D., Orviatt, S., Pittman, J., Smith, I., Chen, L., and Clow, J. (1998). QuickSet: Multimodal interaction for distributed applications. In *Proceedings of The Fifth ACM International Multimedia Conference (MULTIMEDIA '97)*, pages 31-40, New York/Reading. ACM Press/Addison-Wesley.

Emura, Satoru and Susumu Tachi. Compensation of Time Lag Between Actual and Virtual Spaces by Multi-Sensor Integration. *Proceedings of the 1994 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems* (Las Vegas, NV, 2-5 October 1994), 463-469.

Feiner S., The Importance of Being Mobile: Some Social Consequences of Wearable Augmented Reality Systems. *Proc. IWAR '99 (Int. Workshop on Augmented Reality)*, San Francisco, CA, October 20-21, 1999, pp. 145-148.

Feiner, S. and Shamash, A. Hybrid user interfaces: Breeding virtually bigger interfaces for physically smaller computers. *Proc. UIST '91 (ACM Symp. on User Interface Software and Technology)*, Hilton Head, SC, November 11-13, 1991, 9-17.

Feiner, S., MacIntyre, B., and Höllerer, T. (1999). Wearing it out: First steps toward mobile augmented reality systems. In Ohta, Y. and Tamura, H., editors, *Mixed Reality: Merging Real and Virtual Worlds*, pages 363-377. Ohmsha (Tokyo)-Springer Verlag, Berlin.

Feiner, S., MacIntyre, B., and Seligmann, D. Knowledge-Based Augmented Reality. *Communications of the ACM*, Vol. 36(7), 1993, pp. 53-62.

Feiner, S., MacIntyre, B., Haupt, M. and Solomon, E. (1993). Windows on the World: 2D Windows for 3D Augmented Reality. In *Proceedings of UIST '93 - The Sixth Annual Symposium on User Interface Software and Technology*, (pp. 145-156): ACM Press.

Feiner, S., MacIntyre, B., Höllerer, T. and Webster, A. (1997). A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment. In *Proceedings of International Symposium on Wearable Computers (ISWC 97)*, (pp. 74-83): IEEE Computer Society.

Feiner, S., Webster, A., Krueger, T., MacIntyre, B., and Keller, E. Architectural anatomy. In *Presence*, 4(3), Summer 1995, 318-325.

Feiner, S.K. (1996). Adding Insight through Animation in Augmented Reality. In *Proceedings of Computer Animation '96*, (pp. 14-15): IEEE Computer Society Press.

Fisher, S. S. (2001). Environmental media: Linking virtual environments to the physical world. In *Proc. ISMR '01 (Second Int. Symp. on Mixed Reality)*, pages 131-132, Yokohama, Japan.

Fishkin, K. P., Moran, T. P., and Harrison, B. L. (1998). Embodied user interfaces: Towards invisible user interfaces. In *Proc. EHCI '98*, Heraklion, Greece.

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: spatially aware palmtop computers. *Communication of the ACM*, 1993. 36(7): pp. 38-49.

Foxlin, E., Harrington, M., and Pfeifer, G. (1998). Constellation: A wide-range wireless motion-tracking system for augmented reality and virtual set applications. In *Proc. SIGGRAPH '98*, pages 372-378.

Fuchs, H. , State, A. , Pisano, E. D. , Garrett, W. F. , Hirota, G., Livingston, M. , Whitton, M. C. and Pizer, S. M. (1996). Towards performing ultrasound-guided needle biopsies from within a head-mounted display. In *Proceedings of VBC '96: Visualization in Biomedical Computing*, 4th International Conference, (pp. 591-600): Springer-Verlag.

Fuchs, M. Livingston, R. Raskar, D. Colucci, K. Keller, A. State, J. Crawford, P. Rademacher, S. Drake and A. Meyer, H. (1998). Augmented Reality Visualization for Laparoscopic Surgery. In *Proceedings of the First International Conference on Medical Image Computing and Computer-Assisted Intervention*.

Fuhrmann A., D. Schmalstieg, W. Purgathofer: Fast Calibration for Augmented Reality In *Proceedings of ACM Virtual Reality Software & Technology '99 (VRST'99)*, short paper, London, December 20-22, 1999.

Fuhrmann A., G. Hesina, F. Faure, Michael Gervautz Occlusion in Collaborative Augmented Environments. *Virtual Environments'99 (Proc. 5th EUROGRAPHICS Workshop on Virtual Environments)*, pp. 179-190, Springer, Vienna, Austria, May 31-June 1, 1999.

Fuhrmann A., H. Löffelmann, D. Schmalstieg, M. Gervautz: Collaborative Visualization in Augmented Reality. *IEEE Computer Graphics & Applications*, Vol. 18, No. 4, pp. 54-59, IEEE Computer Society, 1998.

Fuhrmann A., H. Löffelmann, D. Schmalstieg: Collaborative Augmented Reality: Exploring Dynamical Systems. In *Proceedings of IEEE Visualization'97*, pp. 459-462, Phoenix, Arizona, Oct. 19-24, 1997.

Furness, T. (1986). The super cockpit and its human factors challenges. In *Proc. Human Factors Society 30th Annual Meeting*, pages 48-52, Santa Monica, CA.

Getting, I. (1993). The global positioning system. *IEEE Spectrum*, 30(12):36-47.

Golding, A. R. and Lesh, N. (1999). Indoor navigation using a diverse set of cheap, wearable sensors. In *Proc. ISWC '99 (Third Int. Symp. on Wearable Computers)*, pages 29-36, San Francisco, CA.

Gottschalk, S. and Hughes, J. (1993). Autocalibration for virtual environments tracking hardware. In *Proc. SIGGRAPH '93*, pages 65-72, Anaheim.

Hoff, W.A. and Nguyen, K. (1996). Computer vision-based registration techniques for augmented reality. In *Proceedings of Intelligent Robots and Computer Vision XV*, (pp. 538-548): SPIE.

Höllerer T., S. Feiner, and J. Pavlik, Situated Documentaries: Embedding Multimedia Presentations in the Real World. *Proc. ISWC '99 (Third Int. Symp. on Wearable Computers)*, San Francisco, CA, October 18-19, 1999, pp. 79-86

Höllerer, T., Feiner, S., Terauchi, T., Rashid, G., and Hallaway, D. (1999). Exploring MARS: Developing indoor and outdoor user interfaces to a mobile augmented reality system. *Computers and Graphics*, 23(6):779-785.

Höllerer, T., Hallaway, D., Tinna, N., and Feiner, S. (2001). Steps toward accommodating variable position tracking accuracy in a mobile augmented reality system. In *2nd Int. Workshop on Artificial Intelligence in Mobile Systems (AIMS '01)*.

Hull, P. Neaves and J. Bedford-Roberts, R. (1997). Towards Situated Computing. In *Proc. ISWC '97 (First Int. Symp. on Wearable Computers)*, pages 146-153, Cambridge, MA.

Ishii, H., Ullmer, B., Tangible bits towards seamless interfaces between people, bits and atoms. *Proceedings of CHI97*. 1997. ACM. pp. 234-241.

Jang, B., Kim, J., Kim, H., and Kim, D. (1999). An outdoor augmented reality system for GIS applications. In Ohta, Y. and Tamura, H., editors, *Mixed Reality, Merging Real and Virtual Worlds*, pages 391-399. Ohmsha/Springer, Tokyo/New York.

Janin, A., Zikan, K., Mizell, D., Banner, M. and Sowizral, H. (1994). A videometric head tracker for augmented reality applications. In *Proceedings of Telem manipulator and Telepresence Technologies*, (pp. 308-315): SPIE.

Janin, Adam L., David W. Mizell, and Thomas P. Caudell. Calibration of Head-Mounted Displays for Augmented Reality Applications. *Proceedings of IEEE VRAIS '93* (Seattle, WA, 18-22 September 1993), 246-255.

Jebara, T., Schiele, B., Oliver, N., and Pentland, A. (1998). DyPERS: Dynamic personal enhanced reality system. In *Proc. 1998 Image Understanding Workshop*, Monterey, CA.

Julier, S., Brown, D., and Baillet, Y. (2001). The need for AI: Intuitive user interfaces for mobile augmented reality systems. In *2nd Int. Workshop on Artificial Intelligence in Mobile Systems (AIMS '01)*.

Julier, S., Lanzagorta, M., Baillet, Y., Rosenblum, L., Feiner, S., Höllerer, T., and Sestito, S. (2000). Information filtering for mobile augmented reality. In *Proc. ISAR '00 (Int. Symposium on Augmented Reality)*, pages 3-11, Munich, Germany.

Julier, Y. Baillet, M. Lanzagorta, D. Brown and L. Rosenblum, S. (2000). BARS: Battlefield Augmented Reality System. In *NATO Symposium on Information Processing Techniques for Military Systems*, Istanbul, Turkey.

Kato, H., Billinghamurst, M. (1999) Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System. In *Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99)*. October, San Francisco, USA.

Kato, H., Billinghamurst, M., Asano, K., Tachibana, K., (2000) An Augmented Reality System and its Calibration based on Marker Tracking. In the *Journal of the Japanese Virtual Reality Society*, Jan 2000.

Kato, H., Billinghamurst, M., Poupyrev, I., Imamoto, K., Tachibana, K., Virtual Object Manipulation on a Table-Top AR Environment. *Proceedings of International Symposium on Augmented Reality (ISAR 2000)*. 2000.

Kaufman, S., Poupyrev, I., Miller, E., Billinghamurst, M., Oppenheimer, P., Weghorst, S. (1997) New 'Augmented Reality' Interface Metaphors for Cardiology Information: The LIMIT Project and an ECG Monitor Object Prototype. *Presented at the American College of Cardiology Scientific Sessions*, Anaheim, CA, March, 1997.

Kaufmann H., D. Schmalstieg, M. Wagner: Construct3D: A Virtual Reality Application for Mathematics and Geometry Education. *Journal of Education and Information Technologies*, Vol. 5, No. 4, pp. 263-276, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000.

Khotake, N., Rekimoto, J., Anzai, Y., InfoStick: an interaction device for Inter-Appliance Computing. *Proceedings of Handheld and Ubiquitous Computing (HUC 99)*. 1999.

Kijima, Ryugo, Eijiroh Yamada, and Takeo Ojika. A Development of Reflex HMD-HMD with Time Delay Compensation Capability. *Proceedings of ISMR 2001* (Yokohama, Japan, 14-15 March 2001), 40-47.

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.

Kiyokawa, K., Takemura, H., Yokoya, N. "A Collaboration Supporting Technique by Integrating a Shared Virtual Reality and a Shared Augmented Reality", *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC '99)*, Vol.VI, pp.48-53, Tokyo, 1999.

Kiyokawa, K., Takemura, H., Yokoya, N. "SeamlessDesign: A Face-to-face Collaborative Virtual / Augmented Environment for Rapid Prototyping of Geometrically Constrained 3-D Objects," *Proceedings of the IEEE International Conference on Multimedia Computing and Systems '99 (ICMCS '99)*, Vol.2, pp.447-453, Florence, 1999.

Klaus, A., Kramer, A., Breen, D., Chevalier, P., Crampton, C., Rose, E., Tuceryan, M., Whitaker, R., Greer, D. (1995) Distributed Augmented Reality for Collaborative Design Applications. In *Proceedings of Eurographics '95*. pp. C-03-C-14, September 1995.

Kutulakos, Kiriakos N. and James R. Vallino. Calibration-Free Augmented Reality. *IEEE Transactions on Visualization and Computer Graphics* 4, 1 (January - March 1998), 1-20.

Lee, S.-W. and Mase, K. (2001). A personal indoor navigation system using wearable sensors. In *Proc. ISMR '01 (Second Int. Symp. on Mixed Reality)*, pages 147-148, Yokohama, Japan.

LeMaster, E. and Rock, S. (2000). Field test results for a self-calibrating pseudolite array. In *Proc. of Institute of Navigation GPS-2000*, Salt Lake City, Utah.

MacIntyre and S. Feiner, B. (1996). Future Multimedia User Interfaces. *Multimedia Systems*, 4(5):250-268.

MacIntyre, B. and Coelho, E. M. (2000). Adapting to dynamic registration errors using level of error (LOE) filtering. In *Proc. ISAR '00 (Int. Symposium on Augmented Reality)*, pages 85-88, Munich, Germany.

Mackay, W., Fayard, A-L, Frobert, L. & Médini, L., (1998) Reinventing the Familiar: Exploring an Augmented Reality Design Space for Air Traffic Control. In *Proceedings of ACM CHI '98 Human Factors in Computing Systems*. Los Angeles, California: ACM/SIGCHI, 1998.

Maes, P., The ALIVE system: wireless, full-body interaction with autonomous agents. *ACM Multimedia Systems*, 1997. 5(2): pp. 105-112.

Mann, S. (1997). Wearable computing: A first step toward personal imaging. *IEEE Computer*, 30(2).

Mark, William, Leonard McMillan, Gary Bishop. Post-Rendering 3D Warping. *Proceedings of 1997 Symposium on Interactive 3D Graphics* (Providence, RI, 27-30 April 1997), 7-16.

Milgram, P. and Kishino, F. (1994, September). A Taxonomy of Mixed Reality Visual Displays. *IEICE Transactions on Information and Systems*, E77-D(9), 1321-1329.

Milgram, P., Takemura, H., Utsumi, A. and Kishino, F. (1994). Augmented reality: a class of displays on the reality-virtuality continuum. In *Proceedings of Telemanipulator and Telepresence Technologies*, (pp. 282-292): SPIE.

Milgram, P., Zhai, S., Drasic, D. and Grodski, J. (1993). Applications of Augmented Reality for Human-Robot Communication. In *Proceedings of IROS '93: the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems, Intelligent Robots for Flexibility*, (pp. 1467-1472): IEEE.

Mizell, D. W. (1994). Virtual reality and augmented reality in aircraft design and manufacturing. In *Proceedings of Wescon/94*, (pp. 91): IEEE.

Murao, M., Arikawa, M., and Okamura, K. (1999). Augmented/reduced spatial hypermedia systems for networked live videos on internet. In *Proc. of Int'l Workshop on Urban Multi-Media/3D mapping (UM3'99)*, pages 15-20, Tokyo Univ.

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.

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.

Olano, Marc, Jon Cohen, Mark Mine, and Gary Bishop. Combating Graphics System Latency. *Proceedings of 1995 Symposium on Interactive 3D Graphics* (Monterey, CA, 9-12 April 1995), 19-24.

Park, A. and Kazman, R. (1994). Augmented Reality for Mining Teleoperation. In *Proceedings of Telemanipulator and Telepresence Technologies*, (pp. 119-129): SPIE.

Piekarski, W., Gunther, B., and Thomas, B. (1999). Integrating virtual and augmented realities in an outdoor application. In *Proc. IWAR '99 (Int. Workshop on Augmented Reality)*, pages 45-54, San Francisco, CA.

Poupyrev, I., Berry, R., Kurumisawa, J., Nakao, K., Billinghamurst, M., et al., Augmented Groove: Collaborative Jamming in Augmented Reality. *Proceedings of SIGGRAPH'2000 Conference Abstracts and Applications*. 2000. ACM. pp. 77.

Poupyrev, I., Billinghamurst, M. Kato, H., May, R.(2000) Integrating Real and Virtual Worlds in Shared Space. In *proceedings of the 5th International Symposium on Artificial Life and Robotics (AROB 5th'00)*, Oita, Japan, 26-28 January, 2000, Vol. 1, pp. 22-25.

Poupyrev, I., Tan, D., Billinghamurst, M., Kato, H., Regenbrecht, H., et al., *Tiles: A Mixed Reality Authoring Interface*. Proceedings of Interact 2001. 2001.

Pouwelse, J., Langendoen, K., and Sips, H. (1999). A feasible low-power augmented reality terminal. In *Proc. IWAR '99 (Int. Workshop on Augmented Reality)*, pages 55-63, San Francisco, CA.

Raskar R., G. Welch, M. Cutts, A. Lake, L. Stesin, H. Fuchs. The office of the future: A unified approach to image-based modeling and spatially immersive displays, *Proc. SIGGRAPH '98*, pp. 179-188, 1998.

Regan, Matthew, and Ronald Pose. Priority Rendering with a Virtual Reality Address Recalculation Pipeline. *Proceedings of SIGGRAPH '94* (Orlando, FL, 24-29 July 1994). In Computer Graphics, Annual Conference Series, 1994, 155-162.

Regan, Matthew, Gavin Miller, Steve Rubin, and Chris Kogelnik. A Real Time Low-Latency Light-Field Renderer. *Proceedings of SIGGRAPH '99* (Los Angeles, CA, 8-13 August 1999), 287-290.

Rekimoto, J. Transvision: A Hand-held Augmented Reality System for Collaborative Design. In *Proceeding of Virtual Systems and Multimedia '96 (VSMM '96)*, Gifu, Japan, 18-20 Sept., 1996.

Rekimoto, J., Ayatsuka, Y., and Hayashi, K. (1998). Augment-able reality: Situated communication through physical and digital spaces. In *Proc. ISWC '98 (Second Int. Symp. on Wearable Computers)*, pages 68-75, Cambridge, MA.

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.

Rekimoto, J., Y. Ayatsuka. CyberCode: Designing Augmented Reality Environments with Visual Tags, *Designing Augmented Reality Environments (DARE 2000)*, 2000.

Rekimoto, Jun, Brygg Ullmer, Haruo Oba. DataTiles: a modular platform for mixed physical and graphical interactions. *Proceedings ACM CHI 2001*, 269 – 276, 2001.

Rekimoto, Jun. NaviCam: A Magnifying Glass Approach to Augmented Reality. *Presence: Teleoperators and Virtual Environments* 6, 4 (August 1997), 399-412.

Rhodes, B. J. (1997). The wearable remembrance agent: A system for augmented memory. In *Proc. ISWC '97 (First Int. Symp. on Wearable Computers)*, pages 123-128, Cambridge, Mass., USA.

Rungarityotin, W. and Starner, T. (2000). Finding location using omnidirectional video on a wearable computing platform. In *Proc. ISWC '00 (Fourth Int. Symp. on Wearable Computers)*, pages 61-68, Atlanta, GA.

Sato, K., Ban, Y., and Chihara, K. (1999). MR aided engineering: Inspection support systems integrating virtual instruments and process control. In Ohta, Y. and Tamura, H., editors, *Mixed Reality, Merging Real and Virtual Worlds*, pages 347-361. Ohmsha/Springer, Tokyo/New York.

Sawada, K., Okihara, M., and Nakamura, S. (2001). A wearable attitude measurement system using a fiber optic gyroscope. In *Proc. ISMR '01 (Second Int. Symp. on Mixed Reality)*, pages 35-39, Yokohama, Japan.

Schmalstieg, D., A. Fuhrmann, G. Hesina: Bridging Multiple User Interface Dimensions with Augmented Reality. In *Proceedings of ISAR 2000*. (Munich, Germany, 5-6 October 2000), 159-164.

Schmalstieg, D., A. Fuhrmann, Z. Szalavari, M. Gervautz: "Studierstube" - An Environment for Collaboration in Augmented Reality. Extended abstract in: *Proceedings of CVE '96 Workshop*. 1996. Paper in: *Virtual Reality - Systems, Development and Applications*, Vol. 3, No. 1, pp. 37-49, 1998.

Schmalstieg, D., L. M. Encarnação, Zs. Szalavári: Using Transparent Props For Interaction With The Virtual Table. *Proceedings of SIGGRAPH Symposium on Interactive 3D Graphics '99*, pp. 147-154, Atlanta, GA, April 26-28, 1999.

Seo, Y. and K.S. Hong. Weakly Calibrated Video-Based Augmented Reality: Embedding and Rendering Through Virtual Camera. *Proceedings of ISAR 2000 (Munich, Germany, 5-6 October 2000)*, 129-136.

So, Richard H. Y. and Michael J. Griffin. Compensating Lags in Head-Coupled Displays Using Head Position Prediction and Image Deflection. *Journal of Aircraft* 29, 6 (November - December 1992), 1064-1068.

Spitzer, M., Rensing, N., McClelland, R., and Aquilino, P. (1997). Eyeglass-based systems for wearable computing. In *Proc. ISWC '97 (First Int. Symp. on Wearable Computers)*, pages 48-51, Cambridge, MA.

Spohrer, J. (1999). Information in places. *IBM Systems Journal*, 38(4):602-628.

Starner, T., Kirsch, D., and Assefa, S. (1997). The locust swarm: An environmentally-powered, networkless location and messaging system. In *Proc. ISWC '97 (First Int. Symp. on Wearable Computers)*, pages 169-170, Cambridge, MA.

Starner, T., Mann, S., Rhodes, B., Levine, J., Healey, J., Kirsch, D., Picard, R., and Pentland, A. (1997). Augmented reality through wearable computing. *Presence*, 6(4):386-398.

State, A., Hirota, G., Chen, D., Garrett, W., and Livingston, M. (1996a). Superior augmented reality registration by integrating landmark tracking and magnetic tracking. In *Proc. SIGGRAPH '96*, pages 429-438, New Orleans, LA.

State, A., Livingston, M., Garrett, W., Hirota, G., Whitton, M., Pisano, E., and Fuchs, H. (1996b). Technologies for augmented reality systems: Realizing ultrasound-guided needle biopsies. In *Proc. SIGGRAPH '96*, pages 439-446, New Orleans, LA.

Sutherland, I. (1968). A head-mounted three dimensional display. In *Proc. FJCC 1968*, pages 757-764, Washington, DC. Thompson Books.

Szalavári Z., E.Eckstein, M. Gervautz. Collaborative Gaming in Augmented Reality. In *Proceedings of VRST'98*, pp.195-204, Taipei, Taiwan, November 2-5, 1998.

Szalavári, Z., Gervautz, M. The Personal Interaction Panel – A Two-Handed Interface for Augmented Reality. *Computer Graphics Forum*, 16, 3 (Proceedings of EUROGRAPHICS'97, Budapest, Hungary), pp. 335-346, September 1997

Thalmann, N.M. and Thalmann, D. (1997). Animating virtual actors in real environments. *Multimedia Systems*, 5(2), 113-125.

Thomas, B., Close, B., Donoghue, J., Squires, J., De Bondi, P., Morris, M., and Piekarski, W. (2000). ARQuake: An outdoor/indoor augmented reality first person application. In *Proc. ISWC '00 (Fourth Int. Symp. on Wearable Computers)*, pages 139-146, Atlanta, GA.

Tuceryan, Mihran, Douglas S. Greer, Ross T. Whitaker, David Breen, Chris Crampton, Eric Rose, and Klaus H. Ahlers. Calibration Requirements and Procedures for Augmented Reality. *IEEE Transactions on Visualization and Computer Graphics* 1, 3 (September 1995), 255-273.

Ullmer B., H. Ishii, D. Glas. mediaBlocks: Physical Containers, Transports, and Controls for Online Media, *Proc. SIGGRAPH '98*, pp. 379-386, July 1998.

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.

Van Craen, R. , De Vlieger, Y. , Vandamme, F. and Vandamme, M. (1995). Company presentations and business data visualization through cooperative virtual and augmented reality. In *Proceedings of 14th International Congress on Cybernetics*, (pp. 1142-1145): Association Internationale Cybernetique.

Van Laerhoven, K. and Cakmakci, O. (2000). What shall we teach our pants? In *Proc. ISWC '00 (Fourth Int. Symp. on Wearable Computers)*, pages 77-83, Atlanta, GA.

Vandamme, F. (1995). V/AR and its application: V/AR as the answer to the Faustian computer dilemma?. In *Proceedings of 14th International Congress on Cybernetics*, (pp. 1119): Association Internationale Cybernetique.

Vandamme, M. and Van Craen, R. (1995). General overview of virtual and augmented reality and its applications. In *Proceedings of 14th International Congress on Cybernetics*, (pp. 1138-1141): Association Internationale Cybernetique.

Walairacht, Somsak, Keita Yamada, Shoichi Hasegawa, Yasuharu Koike, and Makoto Sato. 4+4 Fingers Manipulating Virtual Objects in Mixed Reality Environment. *Proceedings of ISMR 2001 (Yokohama, Japan, 14-15 March 2001)*, 27-34.

Want, R., Schilit, B. N., Adams, N. I., Gold, R., Petersen, K., Goldberg, D., Ellis, J. R., and Weiser, M. (1995). An overview of the PARCTAB ubiquitous computing experiment. *IEEE Personal Communications*, 2(6):28-43.

Ward, M., Azuma, R., Bennett, R., Gottschalk, S., and Fuchs, H. (1992). A demonstrated optical tracker with scalable work area for head-mounted display systems. *Computer Graphics (1992 Symposium on Interactive 3D Graphics)*, 25(2):43-52.

Webster, A., Feiner, S., MacIntyre, B., Massie, W., and Krueger, T. (1996). Augmented reality in architectural construction, inspection and renovation. In *Proc. ASCE Third Congress on Computing in Civil Engineering*, pages 913-919, Anaheim, CA.

Weghorst, S. (1997). Augmented Reality and Parkinson's Disease. *Communications of the ACM*, 40(8), 47-48.

Weiser, M. (1991). The computer for the 21st century. *Scientific American*, 3(265):94-104.

Welch, Greg and Gary Bishop. SCAAT: Incremental Tracking with Incomplete Information. Computer Graphics annual conference series 1997, (*Proceedings of SIGGRAPH '97*) (Los Angeles, CA, 3-8 August 1997), 333-344.

Welch, Greg, Gary Bishop, Leandra Vicci, Stephen Brumback, Kurtis Keller, and D'nardo Colucci (2001). "High-Performance Wide-Area Optical Tracking -The HiBall Tracking System," *Presence: Teleoperators and Virtual Environments* 10(1).

Welch, Greg, Gary Bishop, Leandra Vicci, Stephen Brumback, Kurtis Keller, D'nardo Colucci. 1999. "The HiBall Tracker: High-Performance Wide-Area Tracking for Virtual and Augmented Environments," *Proceedings of the ACM Symposium on Virtual Reality Software and Technology 1999 (VRST 99)*, University College London, December 20-22, 1999.

Wellner, P., Interaction with paper on the digital desk. *Communications of the ACM*, 1993. 36(7): pp. 87-96. Wloka, D. W. (1996). CAVE: Personal or Small Group Non-HMD-based head tracked wrap-around Virtual Environment - The System of the Future for Virtual Reality Applications?. In *Proceedings of Virtual Reality World '96*, (pp. 1-8): Mecklermedia.

Wloka, Matthias M. Lag in Multiprocessor Virtual Reality. *Presence: Teleoperators and Virtual Environments* 4, 1 (Winter 1995), 50-63.

You, Suya, Ulrich Neumann, and Ronald Azuma. Hybrid Inertial and Vision Tracking for Augmented Reality Registration. *Proceedings of IEEE VR '99* (Houston, TX, 13-17 March 1999), 260-267.

You, Suya, Ulrich Neumann, and Ronald Azuma. Orientation Tracking for Outdoor Augmented Reality Registration. *IEEE Computer Graphics and Applications* 19, 6 (Nov/Dec 1999), 36-42.

Selected Papers

Azuma, R. and Bishop, G. (1994). Improving Static and Dynamic Registration in an Optical See-Through HMD. In *Proceedings of SIGGRAPH '94*, (pp. 197-204): ACM SIGGRAPH.

Azuma, Ronald T. A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments* 6, 4 (August 1997), 355 - 385. Earlier version appeared in Course Notes #9: Developing Advanced Virtual Reality Applications, ACM SIGGRAPH '95 (Los Angeles, CA, 6-11 August 1995), 20-1 to 20-38.

Azuma, Ronald, Bruce Hoff, Howard Neely III, Ron Sarfaty. A Motion-Stabilized Outdoor Augmented Reality System. *Proceedings of IEEE VR '99* (Houston, TX, 13-17 March 1999), 252-259.

Billinghurst, M., H. Kato, I. Poupyrev. *Tangible Augmented Reality*. Technical Report, HITLab, 2001.

Billinghurst, 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

Butz, A., Höllerer, T., Feiner, S., MacIntyre, B., Beshers, C. Enveloping Users and Computers in a Collaborative 3D Augmented Reality. In *Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality '99 (IWAR '99)*, October 20-21, San Francisco, CA, 1999, pp. 35-44.

D. Schmalstieg, A. Fuhrmann, G. Hesina, Zs. Szalavari, L. M. Encarnação, M. Gervautz, W. Purgathofer: The Studierstube Augmented Reality Project. *Technical report TR-186-2-00-22*, Vienna University of Technology, December 2000.

Feiner, S., MacIntyre, B., Höllerer, T. and Webster, A. (1997). A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment. In *Proceedings of International Symposium on Wearable Computers (ISWC 97)*, (pp. 74-83): IEEE Computer Society.

Höllerer T., S. Feiner, and J. Pavlik, Situated Documentaries: Embedding Multimedia Presentations in the Real World. *Proc. ISWC '99 (Third Int. Symp. on Wearable Computers)*, San Francisco, CA, October 18-19, 1999, pp. 79-86

Kato, H., Billinghurst, M. (1999) Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System. In *Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99)*. October, San Francisco, USA.

Kato, H., Billinghurst, M., Poupyrev, I., Imamoto, K., Tachibana, K., Virtual Object Manipulation on a Table-Top AR Environment. *Proceedings of International Symposium on Augmented Reality (ISAR 2000)*. (Munich, Germany, 5-6 October 2000), 2000.

Khotake, N., Rekimoto, J., Anzai, Y., InfoStick: an interaction device for Inter-Appliance Computing. *Proceedings of Handheld and Ubiquitous Computing (HUC 99)*. 1999.

Poupyrev, I., Tan, D., Billingham, M., Kato, H., Regenbrecht, H., et al., Tiles: A Mixed Reality Authoring Interface. *Proceedings of Interact 2001*. 2001.

Rekimoto, J., Saitoh, M., Augmented surfaces: A spatially continuous work space for hybrid computing environments. *Proceedings of CHI'99*. 1999. ACM. pp. 378-385.

Rekimoto, Jun. NaviCam: A Magnifying Glass Approach to Augmented Reality. *Presence: Teleoperators and Virtual Environments* 6, 4 (August 1997), 399-412.

Schmalstieg, D., A. Fuhrmann, G. Hesina: Bridging Multiple User Interface Dimensions with Augmented Reality. In *Proceedings of ISAR 2000*. (Munich, Germany, 5-6 October 2000), 159-164.

Schmalstieg, D., A. Fuhrmann, Z. Szalavari, M. Gervautz: "Studierstube" - An Environment for Collaboration in Augmented Reality. Extended abstract in: *Proceedings of CVE '96 Workshop*. 1996. Paper in: *Virtual Reality - Systems, Development and Applications*, Vol. 3, No. 1, pp. 37-49, 1998.

Schmalstieg, D., L. M. Encarnação, Zs. Szalavári: Using Transparent Props For Interaction With The Virtual Table. *Proceedings of SIGGRAPH Symposium on Interactive 3D Graphics '99*, pp. 147-154, Atlanta, GA, April 26-28, 1999.

You, Suya, Ulrich Neumann, and Ronald Azuma. Hybrid Inertial and Vision Tracking for Augmented Reality Registration. *Proceedings of IEEE VR '99* (Houston, TX, 13-17 March 1999), 260-267.

A Motion-Stabilized Outdoor Augmented Reality System

Ronald Azuma, Bruce Hoff, Howard Neely III, Ron Sarfaty
HRL Laboratories
3011 Malibu Canyon Rd; Malibu, CA 90265
{azuma, hoff, neely, rsarfaty}@HRL.com

Abstract

Almost all previous Augmented Reality (AR) systems work indoors. Outdoor AR systems offer the potential for new application areas. However, building an outdoor AR system is difficult due to portability constraints, the inability to modify the environment, and the greater range of operating conditions. We demonstrate a hybrid tracker that stabilizes an outdoor AR display with respect to user motion, achieving more accurate registration than previously shown in an outdoor AR system. The hybrid tracker combines rate gyros with a compass and tilt orientation sensor in a near real-time system. Sensor distortions and delays required compensation to achieve good results. The measurements from the two sensors are fused together to compensate for each other's limitations. From static locations with moderate head rotation rates, peak registration errors are ~2 degrees, with typical errors under 1 degree, although errors can become larger over long time periods due to compass drift. Without our stabilization, even small motions make the display nearly unreadable.

1 Motivation

Several prototype Augmented Reality (AR) systems have been built for *indoor* applications such as medical visualization, manufacturing, and entertainment. Representative examples include [5] and [11]; see [2] for more references. In contrast, hardly any *outdoor* AR applications have been explored. If portable, personal AR systems existed, they would open up new classes of applications. For example, a person navigating outdoors (such as a hiker or a soldier) must manually read compass and GPS measurements, look at a 2-D map, and mentally match that information with what he sees in the surrounding environment. With a personal outdoor AR system, he could instead see spatially-located icons and labels placed directly over the objects of interest, identifying landmarks, the path to travel, and dangerous areas to avoid, all without increasing his cognitive load. Such personal AR systems could also aid distributed, collaborative teams. Widely-dispersed users lack a common context to share spatially-located information.

Radioing other team members to observe the “3rd white building to the right of the church” is useless when they see the town from different vantage points. But personal outdoor AR displays could unambiguously identify the building of interest to each team member. AR may also be a natural interface for some outdoor wearable computer systems, instead of the standard WIMP interfaces that distract the user from the real world rather than complementing it [9].

2 Approach

Outdoor AR applications have rarely been attempted because building an effective outdoor AR system is much more difficult than building an indoor system. First, fewer resources are available outdoors. Computation, sensors and power are limited to what a user can reasonably carry. Second, we have little control over the environment outdoors. In an indoor system, one can carefully control the lighting conditions, select the objects in view, add strategically located fiducials to make the tracking easier, etc. But modifying outdoor locations to that degree is unrealistic, so many existing AR tracking strategies are invalid outdoors. Finally, the range of operating conditions is greater outdoors. The ambient light an outdoor display must operate in ranges from bright sunlight to a moonless night. Environmental ruggedness is vital. Ultimately, we desire the “holy grail” of AR systems: accurate operation indoors, outdoors, and everywhere else.



Figure 1: Virtual labels over outdoor landmarks at Pepperdine University, seen from HRL's Bldg. 250

Our approach is to develop hybrid tracking systems that will move us toward this ultimate goal. No single tracking technology has the performance required to meet the stringent needs of outdoors AR. However,

appropriately combining multiple sensors may lead to a viable solution faster than waiting for any single technology to solve the entire problem. The system described in this paper is our first step in this process. To simplify the problem, we assume the real-world objects are distant (e.g., 500+ meters), which allows the use of differential GPS for position tracking. Then we focus on the largest remaining sources of registration error (misalignments between virtual and real): the dynamic errors caused by lag in the system and distortion in the sensors. Compensating for those errors means *stabilizing* the display against user motion. We do this by a hybrid tracker combining rate gyros with a compass and tilt orientation sensor. This system forms a base from which we will improve, with our collaborators at UNC Chapel Hill, USC, and Raytheon, to strive toward the ultimate goal.

3 Contribution

This system is the first motion-stabilized AR display that works outdoors and achieves tighter registration than any previous outdoor AR system, to our knowledge. Figure 1 shows an example of what our system displays: virtual text labels registered over outdoor landmarks. While our system still has apparent registration errors and limitations on where it can operate outdoors, it still represents a large step forward in outdoor AR tracking.

Our system is most closely related to three sets of previous work. First is Columbia's Touring Machine [6]: an outdoor AR system using a differential GPS and a compass and tilt sensor for tracking. That work focused on potential applications rather than on accurate registration, so the paper does not claim any specific accuracies. Video demos of the Touring Machine show large registration errors (10+ degrees) as the user walks. Our system uses hybrid tracking to stabilize the display, making our registration errors much smaller, and we discuss sensor distortions and calibrations that [6] does not. Second, [1] describes an indoor motion-stabilized display for an optical see-through HMD. While our stabilization strategy is similar, this system differs in the choice of sensors and mathematics required to make this outdoor system work. The different sensors required different calibration and system design decisions. Finally, InterSense builds commercial hybrid trackers. [7] describes an orientation-only sensor that also uses gyros and a compass and tilt sensor. Our work differs in our compensation for sensor distortions, our mathematics, and an actual demonstration and evaluation of registration accuracy in an outdoor AR system. [8] describes an indoor ultrasonic - inertial hybrid tracker but that does not apply to outdoor AR. Concurrent with this work, researchers at the Rockwell Science Center

have been building an AR system that achieves registration by recognizing silhouettes at the visual horizon [4].

4 System

4.1 Overview

Figure 2 shows the system dataflow. Three sets of sensors are used: the Omnistar 7000 differential GPS receiver, a Precision Navigation TCM2 compass and tilt sensor, and three Systron Donner GyroChip II QRS14-500-103 rate gyroscopes (± 500 degrees per second range). The Omnistar provides outputs at 1 Hz, with 2-3 meters typical error, which we verified against a survey marker in the town of Malibu. The TCM2 updates at 16 Hz and claims ± 0.5 degrees of error in yaw. The gyros are analog devices which we sample at 1 kHz using a 16-bit A/D PCMCIA card (National Instruments DAQCard-AI-16XE-50). The other two sensors are read via serial lines. A FutureTech 200 MHz Pentium laptop PC reads the sensors. Section 4.2 describes the sensor distortions and calibration required. The DGPS sensor directly provides the position, but the other two sensor outputs are fused together to determine the orientation, as described in Section 4.3. The user location is then passed to the renderer for display (Section 4.4). The display is a monocular, monochrome optical see-through HMD (Virtual Vision V-Cap) with VGA resolution that we have extensively remounted to provide a rigid relationship between the HMD and the sensors. The entire system renders new images at ~ 60 Hz, matching the update rate of the display hardware. The software is a near real time set of threads and processes running under Windows NT 4.0.

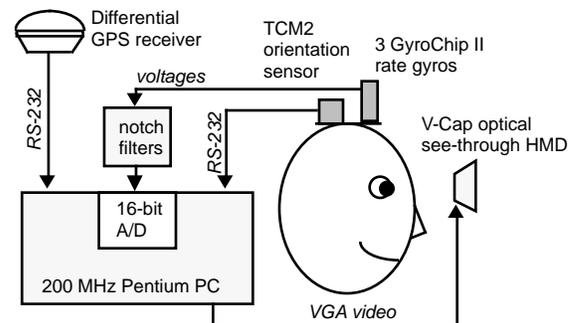


Figure 2: System dataflow

The system operates in both head-worn and hand-held modes. Figure 3 shows the HMD and sensors in a head-worn configuration. However, for ease of demonstrating this system to large groups of people, we also reconfigured the display as a handheld device, with a color video camera (Toshiba IK-M43S) placed where the user's eye normally is, and the video output is shown on a monitor (Figures 4

and 5). We also use the video camera to record the display output, providing the images in this paper.



Figure 3: HMD configuration

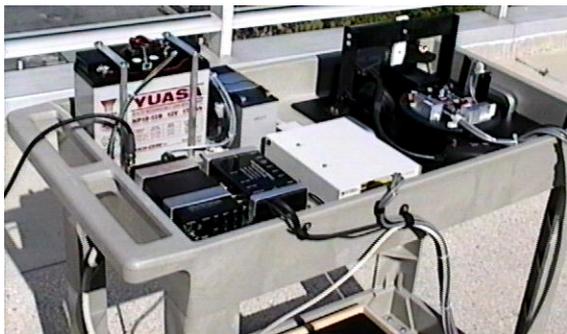


Figure 4: Cart supporting handheld configuration

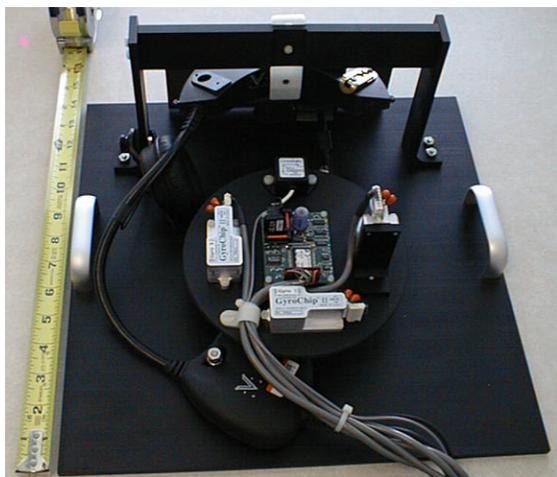


Figure 5: Handheld configuration

4.2 Sensor Calibration

Compass Calibration: We found the TCM2 had significant distortions in the heading output provided by the compass, requiring a substantial calibration effort.

Besides the constant magnetic declination, the compass is affected by local distortions of Earth's magnetic field. We built a non-ferrous mechanical turntable to measure these errors at locations we felt were far from any obvious sources of distortion. Figure 6 shows some collected distortion maps, which were taken in pairs at different locations and times. The distortions have peak-to-peak values of about 2-4 degrees. While the patterns within each pair (which were taken within 30 minutes of each other) are similar, they can be quite different across different pairs. Unfortunately, it is difficult to build a working AR display that does not place some sources of magnetic distortion in the general vicinity of the compass. In the real system, compass errors can have peak-to-peak values of 20-30 degrees.

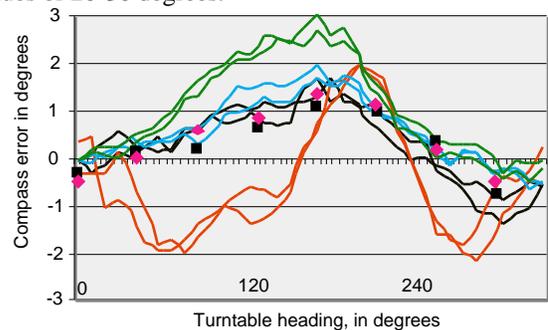


Figure 6: Pairs of compass distortions measured at different locations and times.

Although the distortion pattern is not consistent across all locations and times, the relative consistency between measurements taken 30 minutes apart gave some hope of calibrating the sensor at the beginning of each session. We measure gross distortion maps, like the ones in Figure 6 by sampling the field every 5 degrees (and linearly interpolating). These maps are refined by observing a few known landmarks in the environment. We can compute the true headings to the landmarks by using the known locations of the landmarks and the measured DGPS location of our observation site. These true headings are compared against the measured compass headings. The differences are corrections which are smoothly blended with the original gross distortion map. This provides a correction function mapping compass headings into true headings (on that day, at that site). Although this process requires more manual effort than is desirable, it was necessary to get the best performance we could out of the electronic compass.

Gyroscope Calibration: We measured the bias of each gyroscope by averaging several minutes of output while the gyros were kept still. For scale, we used the specified values in the manufacturer's test sheets for each gyro. The GyroChips have a large internal noise spike around 322 Hz, so we apply active notch filters, designed by Vern Chi

of UNC Chapel Hill, before digitizing the signal [3]. The filters provide over 20 dB of attenuation between 310 and 340 Hz.

To check the bias and scale parameters, we performed an *open-loop* integration of gyroscope output, comparing the integrated values against the rotation measured on a mechanical turntable. After 10-20 seconds of integration, the error was on the order of 0.1 degrees, which gave us confidence in the gyro performance and calibration.

Sensor Latency Calibration: The gyro outputs change quickly in response to user motion, and they are sampled at 1 kHz. In contrast, the TCM2 responds slowly and is read at 16 Hz over a serial line. Therefore, when TCM2 and gyro inputs are read simultaneously, there is some unknown difference in the times of the physical events they each represent.

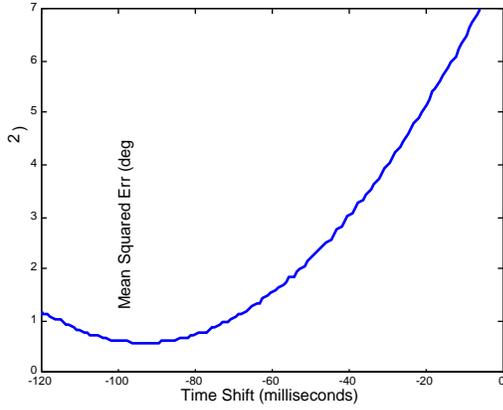


Figure 7: Determining relative latency between TCM2 and gyroscopes

We determined this relative latency in the following way: For several sets of collected TCM2 and gyro data, we integrated the gyro yaw rate to produce heading trajectories. We then integrated the squared difference between the gyro-based heading trajectory and the TCM2-based heading trajectory, varying as a parameter the temporal shift between the two sequences. The result is shown in Figure 7. Across datasets the optimal offset is consistently 92 msec. This latency difference is accounted for in the sensor fusion code.

4.3 Sensor Fusion and Filtering

The goal of sensor fusion is to estimate the angular position and rotation rate of the head from the input of the TCM2 and the three gyroscopes. This position is then extrapolated one frame into the future to estimate the head orientation at the time the image is shown on the see-through display (Figure 8).

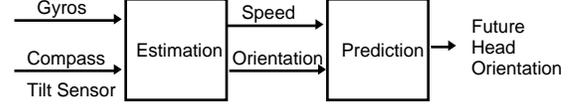


Figure 8: Schematic for fusion and prediction

We relate the kinematic variables of head orientation and speed via a discrete time dynamic system. We define x to be the six dimensional state vector including the three orientation values, as defined for the TCM2 sensor, and the three speed values, as defined for the gyroscopes,

$$x = [r_C \quad p_C \quad h_C \quad r_g \quad p_g \quad h_g]^T$$

where r , p , and h denote roll, pitch, and heading respectively, and the subscripts c and g denote compass (TCM2) and gyroscope, respectively. The first three values are angles and the last three are angular rates. The system is written,

$$x_{i+1} = A_i x_i + w_i, \quad (1)$$

where w_i is noise,

$$A_i = \begin{bmatrix} I_{3 \times 3} & \Delta t A_{12}(x_i) \\ 0_{3 \times 3} & I_{3 \times 3} \end{bmatrix},$$

$$A_{12}(x) = \begin{bmatrix} cpc^2 r/a^2 & asrcr sp(t^2 r + 2/c^2 p) & atpc^2 r/c^2 p \\ 0 & a/cp & -atr \\ 0 & atr/cp & a/c^2 p \end{bmatrix},$$

$$a = \frac{1}{\sqrt{1 + t^2 p + t^2 r}},$$

where $c\theta = \cos(\theta)$, $s\theta = \sin(\theta)$, $t\theta = \tan(\theta)$. For example, $cp = \cos(p)$ and $t^2 r = \tan^2(r)$.

r and p are the TCM2 roll and pitch values (r_c and p_c) in x , and Δt is the time step (1 ms). The matrix A_i comes from the definitions of the TCM2 roll, pitch, heading quantities and the configuration of the gyroscopes. A_{12} translates small rotations in the sensor suite's frame to small changes in the TCM2 variables. The derivation is, unfortunately, too long to include here.

To predict the head orientation one frame into the future, we use a linear motion model: We simply add to the current orientation the offset implied by the estimated rotational velocity. This is done by converting the orientation (the first 3 terms of x) to quaternions and using quaternion multiplication to combine them. We will incorporate more sophisticated predictors in the future.

The fusion of the sensor inputs is done by a filter equation shown below. It gives an estimate of x_i every time step (every millisecond), by updating the previous estimate. It combines the model prediction given by (1) with a correction given by the sensor input. The filter equation is,

$$x_{i+1} = A_i x_i + K_i \left(z_{i+1} - A_i \begin{bmatrix} x_{i-92}^{1-3} \\ x_i^{4-6} \end{bmatrix} \right) \quad (2)$$

where K_i is the gain matrix that weights the sensor input correction term and has the form,

$$K_i = K = \begin{bmatrix} g_c I_{3 \times 3} & 0_{3 \times 3} \\ 0_{3 \times 3} & g_g I_{3 \times 3} \end{bmatrix}.$$

g_c and g_g are scalar gains parameterizing the gain matrix. z_{i+1} is the vector of sensor inputs, where the first 3 terms are the calibrated TCM2 measurements (angles) and the last three are the calibrated gyroscope measurements (angular rates). Since the compass input has a 92 msec latency, the first 3 terms of z_{i+1} are compared not against the first three terms of the most recent estimated state (x_i) but against those terms of the estimate which is 92 msec old. During most time steps there is no TCM2 input. In those cases g_c is set to zero, i.e. there is no sensor correction from the compass input.

A word about optimal filtering: Equation (2) above is in the form of a Kalman filter, where K_i would be the Kalman gain, which is based on the relative noisiness of the sensors and model dynamics and provides the weighting which yields the optimal estimation. We can measure the necessary sensor noise covariance matrices, but the process noise is more elusive. Generally, in the Kalman approach the process noise is assumed to be zero mean, Gaussian and white. These assumptions are likely to be invalid, since driving input is lumped in with the noise in (1). Since we did not have a way to accurately determine the process noise, achieving optimality by tuning a Kalman filter did not appear practical. Additionally, calculating the Kalman gain requires several matrix inversions per time step, which is undesirable for a real-time system with limited computing power. Therefore, we chose the alternative approach of using a *constant* gain matrix K , with a small number of parameters, so that empirical tuning of the gain is tractable. The question then is whether exploration of the parameter space can yield a gain matrix which produces desirable filter behavior.

Numerical optimization of the two parameters, g_c and g_g , would be possible if we had “ground truth” against which to compare filter output. Lacking that, we vary the two parameters, first while running the filter in simulation on collected data, and second while using the filter in the actual system. In simulation we use as ground truth the stable compass reading when the system is still. We also can “see” how jittery the TCM2 output is, and look for “smoother” behavior in the filter output. In the integrated system, we can get an informal “feel” of the quality of the performance by watching how closely labels track landmarks in the environment. (In the future we may

formalize this using video capture techniques.) Each gain can range from 0 to 1. Small gain values indicate trust in the model over the sensors, and large values indicate trust in the sensors over the model. In practice we found that the gyros were very reliable, and set g_g to 1.0, essentially “integrating” the gyro input. The compass provided a small corrective contribution, preventing drift in the long term. We found it sufficient to set g_c to 0.05. A more complicated filter would adaptively change the gain settings, but at least for this initial system we found that constant gains produced satisfactory results.

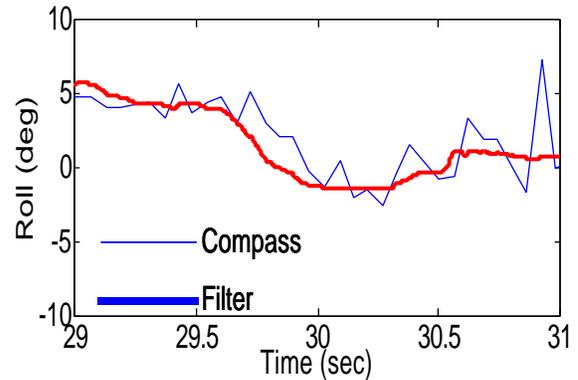


Figure 9: A sequence of roll data. Compass measurements are smoothed by the fusion.

Figures 9, 10, and 11 show graphs of the fusion algorithm’s output, with compass measurements for comparison. Note the filter output leads the compass, due to immediacy of the gyro information, versus the compass’ 92 msec lag. Gyro input also allows the filter to disregard the noisy tilt sensor values. Figure 11 shows output during settling. Estimation quickly reverts to match compass when gyro outputs are zero. Empirically, the chosen fusion algorithm seems to perform well. Quantification of filter accuracy must wait until “ground truth” is available to compare the filter output against.

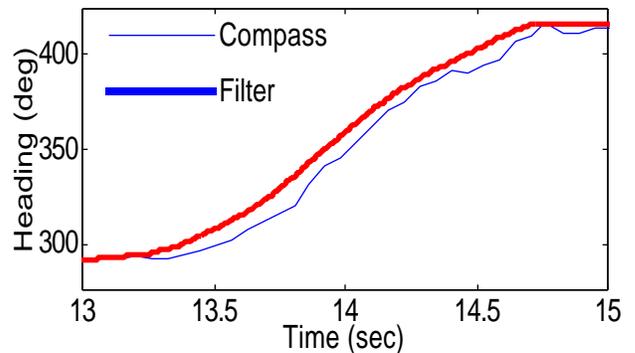


Figure 10: Sequence of heading data. Filter output leads compass measurement.

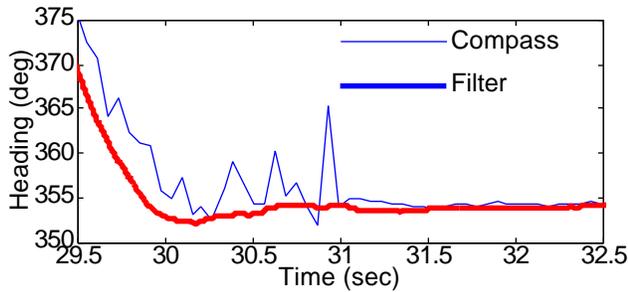


Figure 11: Sequence of heading data. Filter output equals compass when user is still.

This fusion algorithm was influenced by the SCAAT (Single Constraint at a Time) filter [13]. While it is not officially a SCAAT filter, it incorporates individual sensor readings into the filter as they are measured rather than waiting for both types of data to become available. This allows the filter to run at the gyro sampling rate (1 kHz) rather than the compass sampling rate (16 Hz).

4.4 Renderer and Database

The software is primarily organized as one renderer object and one or more database objects. Each database entry stores the Earth-centered (latitude, longitude, altitude) location, Cartesian (x, y, z) location, classification data, and other data for each object. At initialization, an initial user Earth-centered location is established for the current database, from which Cartesian equivalents are created for all current and new database objects, for use in rendering.

The performance design goal for the renderer was to handle a database of up to 50 locations, with up to 10 object labels displayed per frame, while achieving reliable 60 Hz updates. To achieve this goal, all periodic rendering and orientation estimation functions were put within the highest priority thread in the system. In NT4, this is achieved by setting the process class to REALTIME and the thread priority to TIME_CRITICAL. Since: a) this priority is higher than NT4 windows functions, b) our processing completes well within 16 ms, c) thread priorities in REALTIME class processes are not aged, d) we start renderer processing following a voluntary yield of the CPU (thereby assuring a full quantum at the start of execution of the workload) and e) the NT4 Workstation time quantum at maximum boost is 18 ms, we should be assured that the renderer will not be preempted during execution, and will therefore have low periodic variation of execution times and no frame losses. Problems that we encountered with this approach are discussed below.

The renderer uses OpenGL for geometry and DirectDraw 3 for drawing and frame buffer management. Initially, the renderer was a purely OpenGL implementation, but we found that both the Microsoft and SGI OpenGL

implementations draw to a back buffer in system memory, then bit block transfer (bit blt) the back buffer to the display. Poor fill rate performance in the laptop display adapter limited display updates to under 10 Hz. Using DirectDraw avoids this problem by drawing to a back buffer in display memory, then flipping the front and back buffers. Fill rate still limited system performance due to back buffer clearing, performed by bit blt from display memory, even though this hardware function was performed in parallel with estimation processing. We kept OpenGL for geometry processing instead of using Direct3D immediate mode because it was far easier to use.

DirectDraw did not, however, solve all problems. One outstanding problem was in implementing execution of the renderer thread at the desired 60 Hz frame rate. We had expected to be able to trigger rendering on an event that would become signaled on completion of display vertical retrace. We found that DirectX does not support this capability, and no such support is currently planned by Microsoft. We next tried to poll for completion of the buffer flip, suspending the thread using the Win32 Sleep() function when the test failed. This was not entirely successful, because the resolution of Sleep() is the clock resolution, which is 10 ms on our system (it can be 15 ms on other systems) [10]. This would force us to limit all rendering and orientation estimation processing to 6 ms to assure 60 Hz operation.

5 Results

We operated this system at four different geographical locations: the patios of two buildings at HRL Laboratories, Malibu Bluffs park, and Webster Field, MD. The Maryland site was for a DARPA Warfighter Visualization demonstration on June 18, 1998. The system proved robust across different geographical locations and for long operation times. At Webster Field, we ran for five continuous hours. Figures 1, 12, 13 and 14 are static images from videotape recorded at these observation points.

For moderate head rotation rates (under ~100 degrees per second) the largest registration errors we usually observed were ~2 degrees, with average errors being much smaller. The biggest problem was the heading output of the compass sensor drifting with time. The output would drift by as much as 5 degrees over a few hours at Webster Field, requiring occasional recalibration to keep the registration errors under control. The magnetic environment at Webster Field was rather harsh (on a runway near many large metal objects and EM sources) so such errors may not be surprising. Overall, however, errors of under one degree were common, placing the virtual labels close

enough to the real objects to unambiguously identify the landmarks to the user.

The stabilization provided by hybrid tracking was a dramatic improvement, and without such compensation the display is unreadable under even small user motions. We ran the demonstration in three different modes: raw compass, calibrated compass, and stabilized. In raw mode, the tracking is based only upon the output of TCM2, with magnetic declination included. The registration errors are over ten degrees in this mode, due to distortions in Earth's magnetic field caused by the local environment and the equipment needed to support the AR display. Then we show the calibrated compass output (running at 16 Hz, which is the limit of the TCM2). With our static calibration routines, the largest registration errors typically observed are ~2 degrees when the display is kept completely still. However, noise in the compass output makes the labels jump around by 0.5 degrees or more, distracting the user. Worse, even small motions make TCM2 output inaccurate, causing large registration errors. These motions can be as small as the vibrations caused when walking around, or even trying to keep one's head still while the wind is blowing. Adding the gyro outputs in our stabilized, hybrid tracking mode mostly overcomes these problems. The display now runs at 60 Hz and the

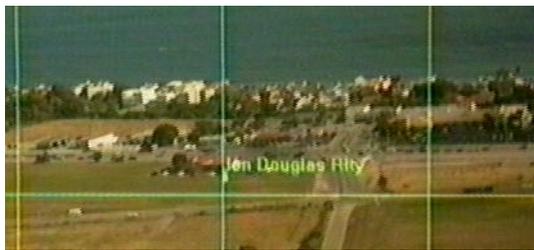


Figure 12: Three Malibu landmarks observed from HRL Bldg. 254 patio (Lifeguard Station, Jon Douglas Realty Building, Serra Retreat Chapel)



Figure 13: View of Pepperdine from Malibu Bluffs (Hornton Administrative Building, Phillips Tower, Landon Center)



Figure 14: Landmarks from Webster Field airstrip (Control tower and windsock)

virtual labels appear to stay with the real landmarks, even as the user moves around. The smoothness is evident in Figure 9 from Section 4.3, which compares the raw compass input against the fusion module output. Comparing these three modes shows the value of the hybrid tracking approach. Without stabilization, the display is virtually unusable under normal operating conditions.

6 Future Work

While this work is a significant step forward in outdoor tracking, it has many problems and limitations. The system as it currently exists is not easily portable. Much can be done to make the equipment smaller and lighter with lower power requirements. Ultimately, MEMS and custom silicon sensors should provide the required performance in miniature packages. Because the system is bulky, we have only operated it at static locations outdoors (once set up, we do not change positions). As the user walks around, we expect that the changing distortion of Earth's magnetic field will pose a serious challenge. The existing static calibration techniques require too much manual operation and must be broadened to handle changing fields.

Adding additional sensors to our hybrid tracker, especially in the visual domain, should help overcome other problems with the base system. We are not able to walk around all locations because GPS is blocked at places that do not have direct line-of-sight to a sufficient number of GPS satellites. And while the current registration errors

may be small enough for some navigation applications, they are still much larger than is desirable and must be further reduced. Fusion of additional sensor inputs should provide the information needed to overcome these problems. In particular, visual input will be an important component [12] [14]. Better prediction and other compensation for sensor and system delays will also further reduce registration errors.

The display is not bright enough to see in bright sunlight, so we use dark translucent plastic to reduce the ambient light that reaches the display (much like a pair of sunglasses). We anticipate that future displays will be bright enough to handle the contrast outdoors.

Finally, future systems require greater attention to environmental ruggedness and ergonomic issues.

Acknowledgments

This work was mostly funded by DARPA ETO Warfighter Visualization, contract N00019-97-C-2013. Our collaborators on this contract are Gary Bishop, Vern Chi and Greg Welch (UNC Chapel Hill), Ulrich Neumann and Suya You (USC), and Rich Nichols and Jim Cannon (Raytheon). We thank Mike Daily for his support and suggestions, and Vern Chi and Gary Bishop for their notch filter design. We also thank the reviewers for their comments and suggestions.

References

- [1] Azuma, Ronald and Gary Bishop. Improving Static and Dynamic Registration in an Optical See-Through HMD. *Proceedings of SIGGRAPH '94* (Orlando, FL, 24-29 July 1994), 197-204.
- [2] Azuma, Ronald T. A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments* 6, 4 (August 1997), 355-385.
- [3] Bainter, James R. Active Filter Has Stable Notch, and Response Can Be Regulated. *Electronics* (Oct. 2, 1975), 115-117.
- [4] Behringer, Reinhold. Improving the Registration Precision by Visual Horizon Silhouette Matching. *Proceedings of the First IEEE Workshop on Augmented Reality* (San Francisco, CA, 1 November 1998).
- [5] Curtis, Dan, David Mizell, Peter Gruenbaum, and Adam Janin. Several Devils in the Details: Making an AR App Work in the Airplane Factory. *Proceedings of the First IEEE Workshop on Augmented Reality* (San Francisco, CA, 1 November 1998).
- [6] Feiner, Steven, Blair MacIntyre, and Tobias Höllerer. A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment. *Proceedings of First International Symposium on Wearable Computers* (Cambridge, MA, 13-14 October 1997), 74-81.
- [7] Foxlin, Eric. Inertial Head-Tracker Sensor Fusion by a Complementary Separate-Bias Kalman Filter. *Proceedings of VRAIS '96* (Santa Clara, CA, 30 March - 3 April 1996), 185-194.
- [8] Foxlin, Eric, Mike Harrington, and George Pfeiffer. Constellation: A Wide-Range Wireless Motion-Tracking System for Augmented Reality and Virtual Set Applications. *Proceedings of SIGGRAPH '98* (Orlando, FL, 19-24 July 1998), 371-378.
- [9] Rhodes, Bradley. WIMP Interface Considered Fatal. Position paper at *IEEE VRAIS '98 Workshop on Interfaces for Wearable Computers* (Atlanta, GA, 15 March 1998).
- [10] Solomon, David A. *Inside Windows NT, Second Edition*. Microsoft Press, 1998.
- [11] State, Andrei, Gentaro Hirota, David T. Chen, Bill Garrett, and Mark Livingston. Superior Augmented Reality Registration by Integrating Landmark Tracking and Magnetic Tracking. *Proceedings of SIGGRAPH '96* (New Orleans, LA, 4-9 August 1996), 429-438.
- [12] Welch, Gregory. Hybrid Self-Tracker: An Inertial / Optical Hybrid Three-Dimensional Tracking System. UNC Chapel Hill Dept. of Computer Science Technical Report TR95-048 (1995), 21 pages.
- [13] Welch, Greg and Gary Bishop. SCAAT: Incremental Tracking with Incomplete Information. *Proceedings of SIGGRAPH '97* (Los Angeles, CA, 3-8 August 1997), 333-344.
- [14] You, Suya, Ulrich Neumann, and Ronald Azuma. Hybrid Inertial and Vision Tracking for Augmented Reality Registration. *Proceedings of IEEE VR '99* (Houston, TX, 13-17 March 1999).

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

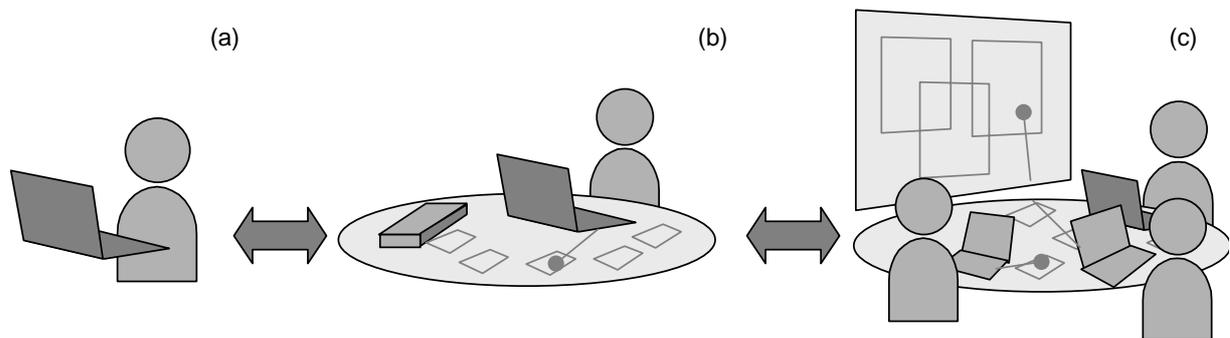


Figure 1: Evolution of spatially continuous workspaces: (a) A user can perform individual tasks with a portable computer. (b) The table becomes an extension of the portable computer. (c) Pre-installed computer displays (table and wall) also serve as shared workspaces for collaborative tasks.

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

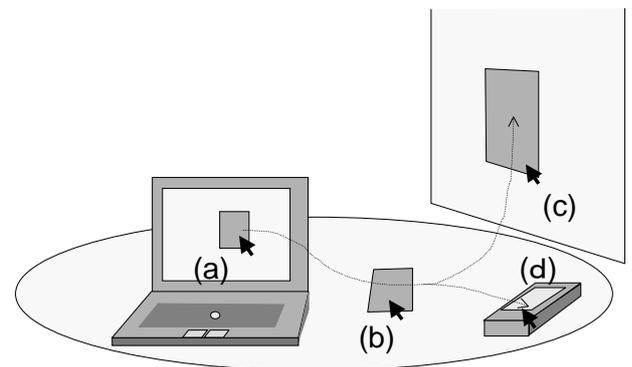


Figure 2: Hyperdragging: A spatially continuous interaction technique for moving information between computers. (a) A user can start moving an object on a computer in the normal manner by dragging it with the pointing device. (b) When the cursor reaches the edge of the screen, it "jumps" to the table 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

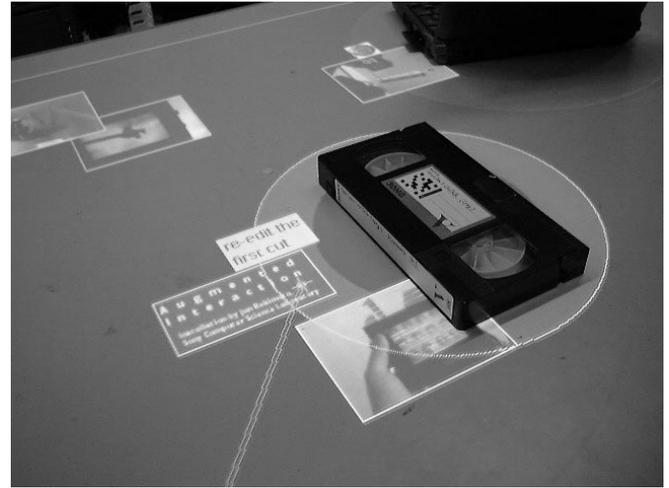


Figure 6: A recognized object (a VCR tape) with an "object aura": A user can attach a digital item by dropping it onto the object aura.

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),

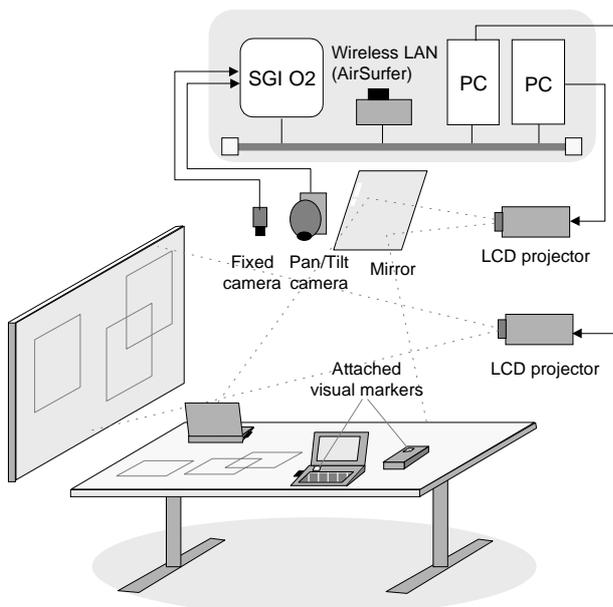


Figure 7: System configuration

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.

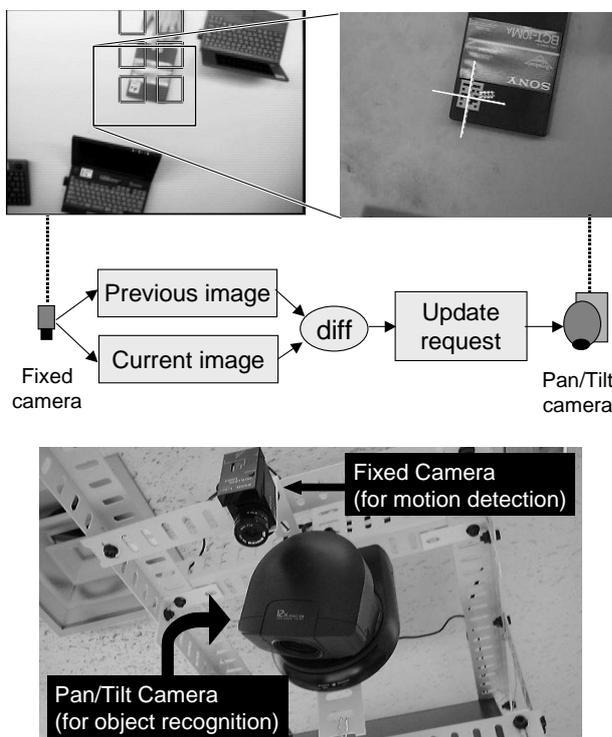


Figure 8: DeskSat uses a combination of two cameras for object recognition

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

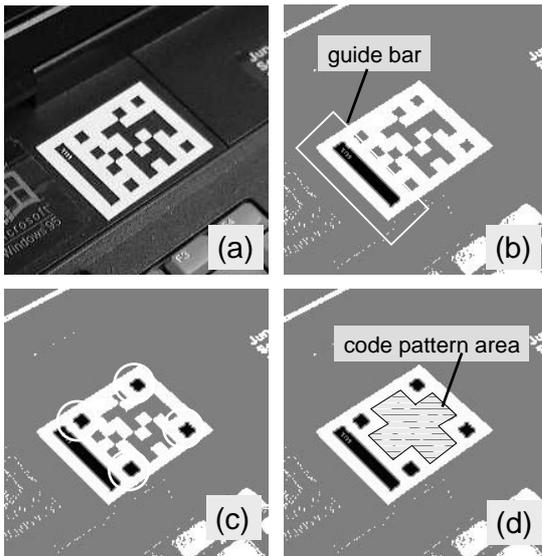


Figure 10: The visual marker recognition algorithm: (a) Original image. (b) Binarized image. Connected regions that have the specific second-order moment are selected. These regions become candidates of a guide bar of the marker. (c) Four corners of the marker region are searched based-on the guide bar 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.

The World through the Computer: Computer Augmented Interaction with Real World Environments

Jun Rekimoto and Katashi Nagao
Sony Computer Science Laboratory Inc.
Takanawa Muse Building,
3-14-13, Higashi-gotanda, Shinagawa-ku,
Tokyo 141 Japan
{rekimoto,nagao}@csl.sony.co.jp
<http://www.csl.sony.co.jp/person/{rekimoto,nagao}.html>

ABSTRACT

Current user interface techniques such as WIMP or the desk-top metaphor do not support real world tasks, because the focus of these user interfaces is only on human-computer interactions, not on human-real world interactions. In this paper, we propose a method of building computer augmented environments using a situation-aware portable device. This device, called *NaviCam*, has the ability to recognize the user's situation by detecting color-code IDs in real world environments. It displays situation sensitive information by superimposing messages on its video see-through screen. Combination of ID-awareness and portable video-see-through display solves several problems with current ubiquitous computers systems and augmented reality systems.

KEYWORDS: user-Interface software and technology, computer augmented environments, palmtop computers, ubiquitous computing, augmented reality, barcode

INTRODUCTION

Computers are becoming increasingly portable and ubiquitous, as recent progress in hardware technology has produced computers that are small enough to carry easily or even to wear. However, these computers, often referred to as PDAs (Personal Digital Assistant) or palmtops, are not suitable for traditional user-interface techniques such as the desk-top metaphor or the WIMP (window, icon, mouse, and a pointing device) interface. The fundamental limitations of GUIs can be summarized as follows:

Explicit operations GUIs can reduce the cognitive overload of computer operations, but do not reduce the volume of operations themselves. This is an upcoming problem for portable computers. As users integrate their computers into their daily lives, they tend to pay less attention to them. Instead, they prefer interacting with each other, and with objects in the real world. The user's focus of interest is not the human-

Permission to make digital/hard copies of all or part of this material for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication and its date appear, and notice is given that copyright is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires specific permission and/or fee.

UIST 95 Pittsburgh PA USA

© 1995 ACM 0-89791-709-x/95/11..\$3.50

computer interactions, but the human-real world interactions. People will not wish to be bothered by tedious computer operations while they are doing a real world task. Consequently, the reduction of the amount of computer manipulation will become an issue rather than simply how to make existing manipulations easier and more understandable.

Unaware of the real world situations Portability implies that computers will be used in a variety of situations in the real world. Thus, dynamical change of functionalities will be required for mobile computers. Traditional GUIs are not designed for such a dynamic environment. Although some context sensitive interaction is available on GUIs, such as *context sensitive help*, GUIs cannot deal with real world contexts. GUIs assume an environment composed of desk-top computers and users at a desk, where the real world situation is less important.

Gaps between the computer world and the real world Objects within a database, which is a computer generated world, can be easily related, but it is hard to make relations among real world objects, or between a real object and a computer based object. Consider a system that maintains a document database. Users of this system can store and retrieve documents. However, once a document has been printed out, the system can no longer maintain such an output. It is up to the user to relate these outputs to objects still maintained in the computer. This is at the user's cost. We thus need computers that can understand real world events, in addition to events within the computer.

Recently, a research field called *computer augmented environments* has been emerged to address these problems [18]. In this paper, we propose a method to build a computer augmented environment using a portable device that has an ability to recognize a user's situation in the real world. A user can see the world through this device with computer augmented information regarding that situation. We call this interaction style *Augmented Interaction*, because this device enhances the ability of the user to interact with the real world environment.

This paper is organized as follows. In the next section, we briefly introduce the idea of proposed interaction style. The

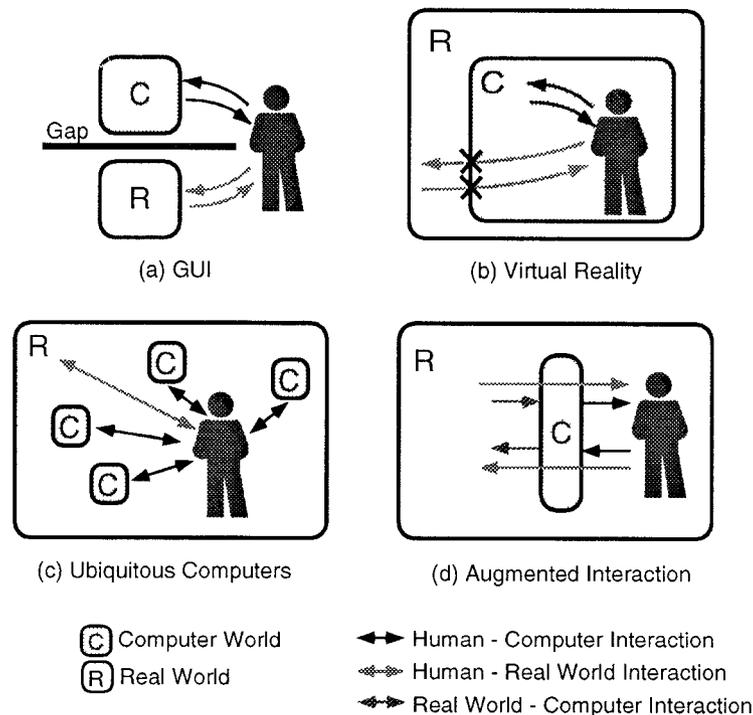


Figure 1: A comparison of HCI styles

following three sections present the NaviCam system, its applications, and its implementation issues. Comparison to other work and our future plans are also discussed in the RELATED WORK section and the FUTURE DIRECTIONS section, respectively.

SITUATION AWARENESS AND AUGMENTED INTERACTION

Augmented Interaction is a style of human-computer interaction that aims to reduce computer manipulations by using environmental information as implicit input. With this style, the user will be able to interact with a real world augmented by the computer's synthetic information. The user's situation will be automatically recognized by using a range of recognition methods, that will allow the computer to assist the user without having to be directly instructed to do so. The user's focus will thus not be on the computer, but on the real world. The computer's role is to assist and enhance interactions between humans and the real world. Many recognition methods can be used with this concept. Time, location, and object recognition using computer vision are possible examples. Also, we can make the real world more understandable to computers, by putting some marks or IDs (bar-codes, for example) on the environment.

Figure 1 shows a comparison of HCI styles involving human-computer interaction and human-real world interaction.

(a) In a desk-top computer (with a GUI as its interaction style), interaction between the user and the computer is isolated from the interaction between the user and the real world. There is a gap between the two interactions. Some researchers are trying to bridge this gap by merging a real desk-top with a desk-top in the computer [12, 17]. (b) In a virtual reality

system, the computer surrounds the user completely and interaction between the user and the real world vanishes. (c) In the ubiquitous computers environment, the user interacts with the real world but can also interact with computers embodied in the real world. (d) Augmented Interaction supports the user's interaction with the real world, using computer augmented information. The main difference between (c) and (d) is the number of computers. The comparison of these two approaches will be discussed later in the RELATED WORK section.

NAVICAM

As an initial attempt to realize the idea of Augmented Interaction, we are currently developing a prototype system called *NaviCam* (NAVIGATION CAMERA). NaviCam is a portable computer with a small video camera to detect real-world situations. This system allows the user to view the real world together with context sensitive information generated by the computer.

NaviCam has two hardware configurations. One is a palmtop computer with a small CCD camera, and the other is a head-up display with a head-mounted camera (Figure 2). Both configurations use the same software. The palmtop configuration extends the idea of position sensitive PDAs proposed by Fitzmaurice [9]. The head-up configuration is a kind of *video see-through HMD* [2], but it does not shield the user's real sight. Both configurations allow the user to interact directly with the real world and also to view the computer augmented view of the real world.

The system uses color-codes to recognize real world situations. The color-code is a sequence of color stripes (red or blue) printed on paper that encodes an ID of a real world

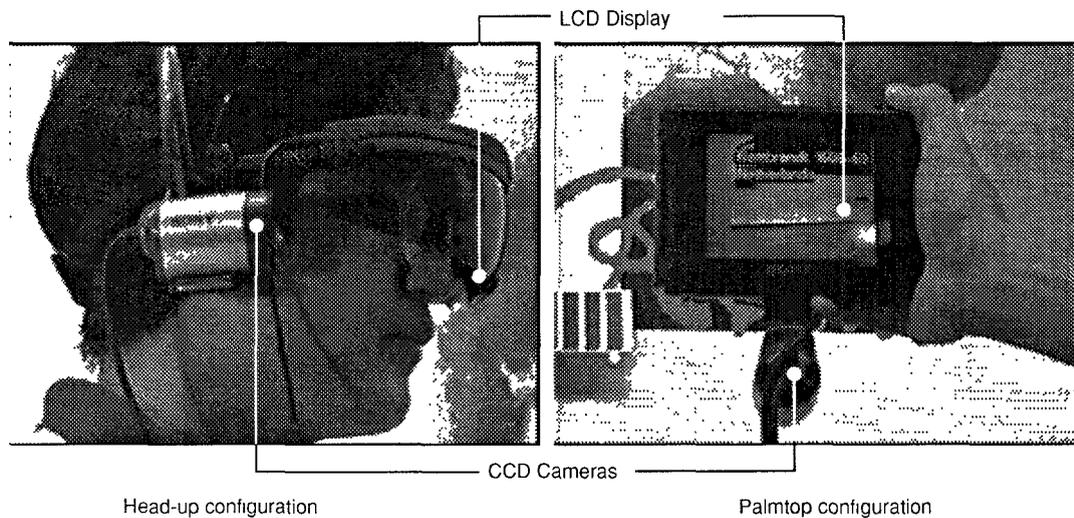


Figure 2: Palmtop configuration and head-up configuration

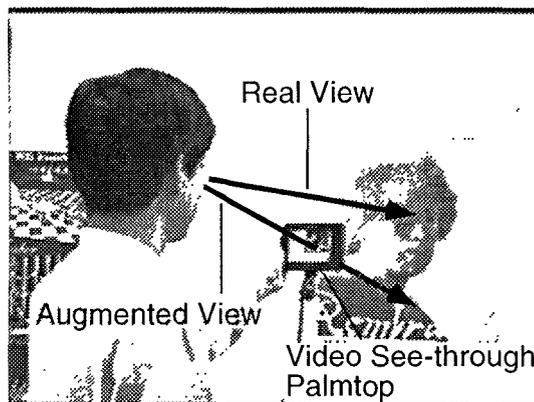


Figure 3: The magnifying glass metaphor

object. For example, the color-code on the door of the office identifies the owner of the office. By detecting a specific color-code, NaviCam can recognize where the user is located in the real world, and what kind of object the user is looking at. Figure 5 shows the information flow of this system. First, the system recognizes a color-code through the camera. Image processing is performed using software at a rate of 10 frames per second. Next, NaviCam generates a message based on that real world situation. Currently, this is done simply by retrieving the database record matching the color-coded ID. Finally, the system superimposes a message on the captured video image.

Using a CCD camera and an LCD display, the palmtop NaviCam presents the view at which the user is looking as if it is a transparent board. We coined the term *magnifying glass metaphor* to describe this configuration (Figure 3). While a real magnifying glass optically enlarges the real world, our system enlarges it in terms of *information*. Just as with a real magnifying glasses, it is easy to move NaviCam around in the environment, to move it toward an object, and to compare the real image and the information-enhanced image.

APPLICATIONS

We are currently investigating the potential of augmented interaction using NaviCam. There follows some experimental applications that we have identified.

Augmented Museum



Figure 4: NaviCam generates information about Rembrandt

Figure 4 shows a sample snapshot of a NaviCam display. The system detects the ID of a picture, and generates a description of it. Suppose that a user with a NaviCam is in a museum and looking at a picture. NaviCam identifies which picture the user is looking at and displays relevant information on the screen. This approach has advantages over putting an explanation card beside a picture. Since NaviCam is a computer, it can generate personalized information depending on the user's age, knowledge level, or preferred language. Contents of explanation cards in today's museums are often too basic for experts, or too difficult for children or overseas visitors. NaviCam overcomes this problem by displaying information appropriate for the owner.

Active Paper Calendar

Figure 6 shows another usage of NaviCam. By viewing a calendar through NaviCam, you can see your own personal

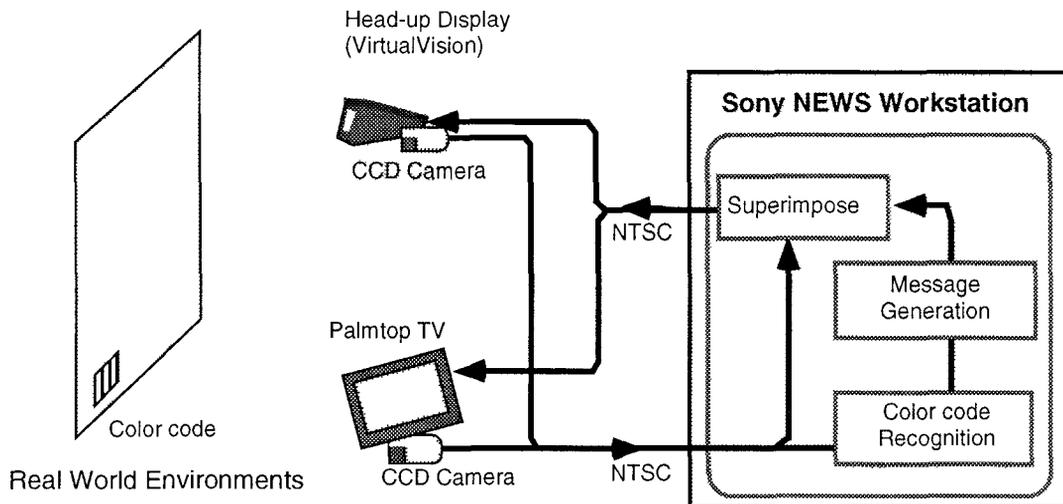


Figure 5: The system architecture of NaviCam

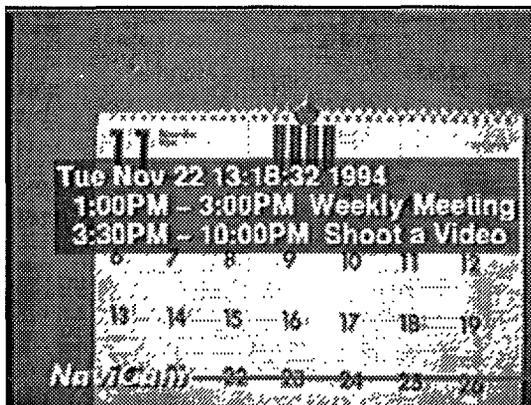


Figure 6: Viewing a paper calendar through NaviCam

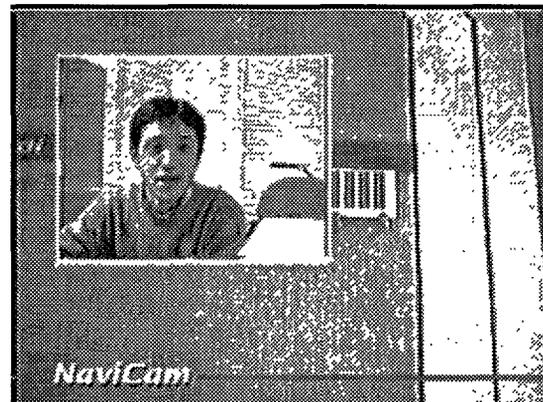


Figure 7: A pseudo-active office door greets a visitor

schedule on it. This is another example of getting situation specific and personalized information while walking around in real world environments. NaviCam can also display information shared among multiple users. For example, you could put your electronic annotation or voice notes on a (real) bulletin board via NaviCam. This annotation can then be read by other NaviCam equipped colleagues.

Active Door

The third example is a NaviCam version of the active door (Figure 7). This office door can tell a visitor where the occupier of the office is currently, and when he/she will come back. The system also allows the office occupier to leave a video message to be displayed on arrival by a visitor (through the visitor's NaviCam screen). There is no need to embed any computer in the door itself. The door only has a color-code ID on it. It is, in fact, a passive-door that can behave as an active-door.

NaviCam as a collaboration tool

In the above three examples, NaviCam users are individually assisted by a computer. NaviCam can also function as a collaboration tool. In this case, a NaviCam user (an operator) is supported by another user (an instructor) looking at the same

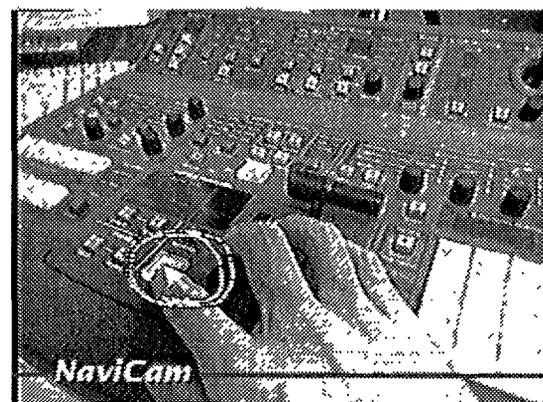


Figure 8: NaviCam can be used as a collaboration tool

screen image from probably a remote location. Unlike other video collaboration tools, the relationship between the two users is not symmetric, but asymmetric. Figure 8 shows an example of collaborative task (video console operation). The instructor is demonstrating which button should be pressed by using a mouse cursor and a circle drawn on the screen. The instructor augments the operator's skill using NaviCam.

Ubiquitous Talker: situated conversation with NaviCam

We are also developing an extended version of NaviCam that allows the user to operate the system with voice commands, called *Ubiquitous Talker*. Ubiquitous Talker is composed of the original NaviCam and a speech dialogue subsystem (Figure 11). The speech subsystem has speech recognition and voice synthesis capabilities. The NaviCam subsystem sends the detected color code ID to the speech subsystem. The speech subsystem generates a response (either voice or text) based on these IDs and spoken commands from the user. The two subsystems communicate with each other through Unix sockets.

An experimental application developed using Ubiquitous Talker is called the *augmented library*. In this scenario, Ubiquitous Talker acts as a personalized library catalogue. The user carries the NaviCam unit around the library and the system assists the user to find a book, or answers questions about the books in the library (Figure 9).

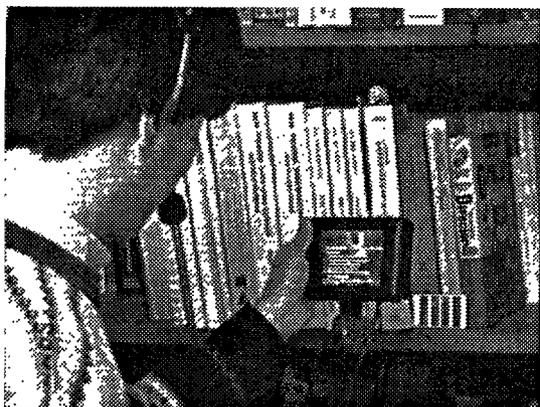


Figure 9: Ubiquitous Talker being used as a library guide

Ubiquitous Talker would also be an important application in the AI research area. Recognizing dialogue contexts remains one of the most difficult areas in natural language understanding. Real-world awareness allows a solution to this problem. For example, the system can respond to a question such as "Where is the book entitled Multimedia Applications?" by answering "It is on the bookshelf *behind* you.", because the system is aware of which bookshelf the user is looking at. It is almost impossible to generate such a response without using real world information. The system also allows a user to use deictic expressions such as "*this* book", because the situation can resolve ambiguity. This feature is similar to multi-modal interfaces such as Bolt's *Put-That-There* system [4]. The unique point in our approach is to use real world situations, other than commands from the user, as a new modality in the human-computer interaction.

For a more detailed discussion of Ubiquitous Talker's natural language processing, please refer to our companion paper [13].

IMPLEMENTATION DETAILS

At this stage, the wearable part of the NaviCam system is connected to a workstation by two NTSC cables and the actual processing is done by the workstation. The workstation component is an X-Window client program written in C. What appears on the palmtop TV is actually an X-window displaying a video image. Video images are transmitted from the video capturing board by using DMA (direct memory access), processed in the system, and sent to the X-Window through the shared-memory transport extension to X.

The following are some of the software implementation issues.

Color code detection

The system seeks out color codes on incoming video images. The image processing is done by software. No special hardware is required apart from video capturing.

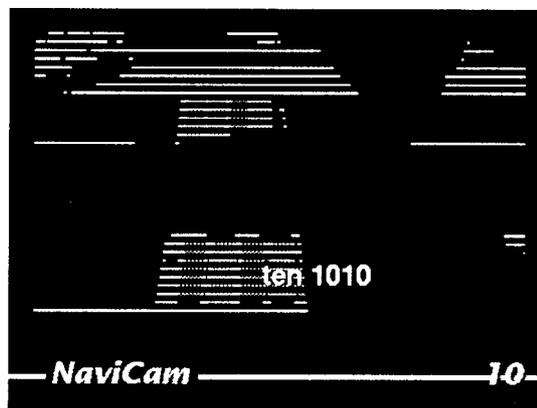


Figure 10: Detecting a color code: a snapshot of what the system is really seeing

The color-code detection algorithm is summarized as follows. First, the system samples some scan lines from the video image (Figure 10). To locate any red and blue bands, each pixel in the scan line is filtered by a color detecting function based on its Y (brightness), R (red) and B (blue) values. Any color bands detected become candidates for a color code. We use the following equations to extract red and blue pixels:

$$\begin{cases} C_1 Y + C_2 < Y - 3R < C_3 Y + C_4 \\ C_5 Y + C_6 < Y - 3B < C_7 Y + C_8 \end{cases} \quad (1)$$

where $Y = R + G + B$, and C_1, \dots, C_8 are constant values. These constants are calculated from sampled pixel values of color-bar images under various lighting conditions. A pixel that satisfies equation 1 is taken as a red pixel. To detect blue pixel, another set of constants (C'_1, \dots, C'_8) is used.

Next, the system selects the most appropriate candidate as the detected color code. Final selection is based on checks for consistency of distance between the color bands. The detected code is then used to generate information on the screen.

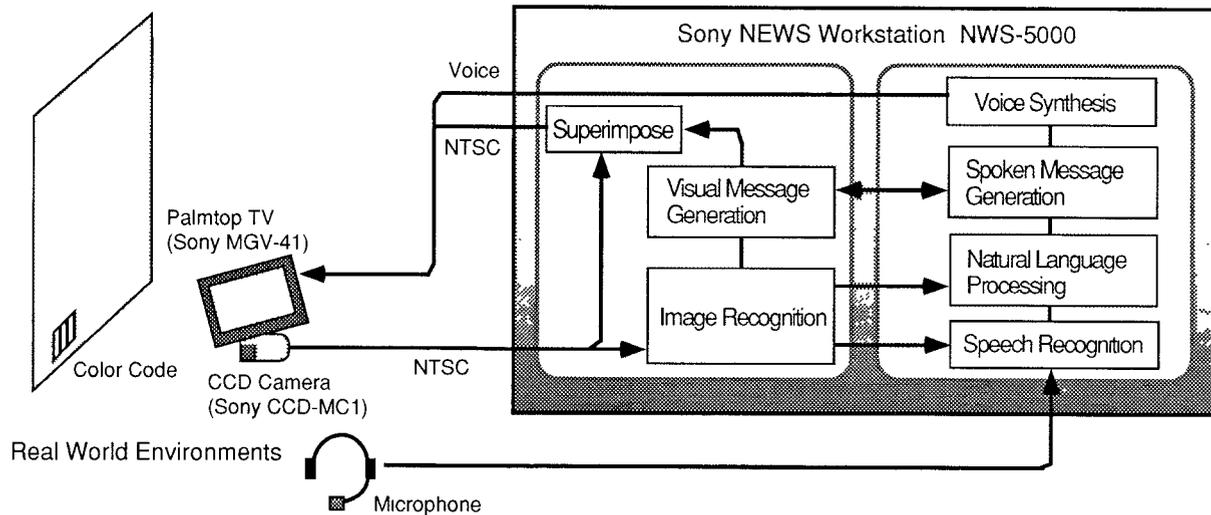


Figure 11: The architecture of Ubiquitous Talker

Using above algorithm, the system can recognize 4-bit color-code IDs (3cm × 5cm in size) at a distance of 30cm – 50cm using the consumer-based small CCD camera (Sony CCD-MC1). IDs are placed in various environments (e.g., offices, libraries, video studios) so the lighting condition also changes depends on the place and the time. Even under such conditions, the color-detecting algorithm was quite robust and stable. This is because equation 1 compensates an effect on pixel values when lighting condition changes.

Superimposing information on a video image

The system superimposes a generated message on the existing video image. This image processing is also achieved using software. We could also use chromakey hardware, but the performance of the software based superimposition is satisfactory for our purposes, even though it cannot achieve a video-frame rate. The message appears near the detected color code on the screen, to emphasize the relation between cause and effect.

We use a 4-inch LCD screen and pixel resolution is 640 × 480. The system can display any graphic elements and characters as the X-Window does. However, it was very hard, if not impossible, to read small fonts through this LCD screen. Currently, we use 24-dot or 32-dot font to increase readability. The system also displays a semi-transparent rectangle as a background of a text item. It retains readability even when the background video image (real scene) is complicated.

Database registration

For the first three applications explained in the APPLICATIONS section, the system first recognizes IDs in the real world environment, then determines what kind of information should be displayed. Thus, the database supporting the NaviCam is essential to the generation of adequate information. The current implementation of the system adopts very simplified approach to this. The system contains a group of command script files with IDs. On receipt of a valid ID, the system invokes a script having the same ID. The invoked script generates a string that appears on the screen. This mechanism works well enough, especially at the prototype stage.

However, we obviously need to enhance this element, before realizing more complicated and practical applications.

RELATED WORK

In this section, we discuss our Augmented Interaction approach in relation to other related approaches.

Ubiquitous computers

Augmented Interaction has similarities to Sakamura's *highly functionally distributed system* (HFDS) concept [14], his TRON house project, and *ubiquitous computers* proposed by Weiser [16]. These approaches all aim to create a computer augmented *real* environment rather than building a *virtual* environment in a computer. The main difference between ubiquitous computing and Augmented Interaction is in the approach. Augmented Interaction tries to achieve its goal by introducing a portable or wearable computer that uses real world situations as implicit commands. Ubiquitous computing realizes the same goal by spreading a large number of computers around the environment.

These two approaches are complementary and can support each other. We believe that in future, human existence will be enhanced by a mixture of the two; ubiquitous computers embodied everywhere, and a portable computer acting as an intimate assistant.

One problem with using ubiquitous computers is reliability. In a ubiquitous computers world, each computer has a different functionality and requires different software. It is essential that they collaborate with each other. However, if our everyday life is filled with a massive number of computers, we must anticipate that some of them will not work correctly, because of hardware or software troubles, or simply because of their dead batteries. It can be very difficult to detect such problem among so many computers and then fix them. Another problem is cost. Although the price of computers is getting down rapidly, it is still costly to embed a computer in every document in an office, for example.

In contrast to ubiquitous computers, NaviCam's situation aware

approach is a low cost and potentially more reliable alternative to embedding a computer everywhere. Suppose that every page in a book had a unique ID (e.g. bar-code). When the user opens a page, the ID of that page is detected by the computer, and the system can supply specific information relating to that page. If the user has some comments or ideas while reading that page, they can simply read them out. The system will record the voice information tagged with the page ID for later retrieval. This scenario is almost equivalent to having a computer in every page of a book but with very little cost. ID-awareness is better than ubiquitous computers from the viewpoint of reliability, because it does not require batteries, does not consume energy, and does not break down.

Another advantage of an ID-awareness approach is the possibility of incorporating existing ID systems. Today, barcode systems are in use everywhere. Many products have barcodes for POS use, while many libraries use a barcode system to manage their books. If NaviCam can detect such commonly used IDs, we should be able to take advantage of computer augmented environments long before embodied computers are commonplace.

Augmented Reality

Augmented reality (AR) is a variant of virtual reality that uses see-through head mounted displays to overlay computer generated images on the user's real sight [15, 8, 6, 2, 7, 5].

AR systems currently developed use only locational information to generate images. This is because the research focus of AR is currently on implementing correct registration of 3D images on a real scene [1, 3]. However, by incorporating other external factors such as real world IDs, the usefulness of AR should be much more improved.

We have built NaviCam in both head-up and palmtop configurations. The head-up configuration is quite similar to other AR systems, though currently NaviCam does not utilize locational information. We thus have experience of both head-up and palmtop type of augmented reality systems and have learned some of the advantages and disadvantages of both.

The major disadvantage of a palmtop configuration is that it always requires one hand to hold the device. Head-up NaviCam allows for hands-free operation. Palmtop NaviCam is thus not suitable for some applications requiring two handed operation (e.g. surgery). On the other hand, putting on head-up gear is, of course, rather cumbersome and under some circumstances might be socially unacceptable. This situation will not change until head-up gear becomes as small and light as bifocal spectacles are today.

For the ID detection purpose, head-up NaviCam is also somewhat impractical because it forces the user to place their head very close to the object. Since hand mobility is much quicker and easier than head mobility, palmtop NaviCam appears more suitable for browsing through a real world environment.

Another potential advantage of the palmtop configuration is that it still allows traditional interaction techniques through its screen. For example, you could to annotate the real world with letters or graphics directly on the NaviCam screen with your finger or a pen. You could also operate NaviCam by

touching a menu on the screen. This is quite plausible because most existing palmtop computers have a touch-sensitive, pen-aware LCD screen. On the other hand, a head-up configuration would require other interaction techniques with which users would be unfamiliar.

Returning to the magnifying glass analogy, we can identify uses for head-up magnifying glasses for some special purposes (e.g. watch repair). The head-up configuration therefore has advantages in some areas, however, even in these fields hand-held magnifying lenses are still dominant and most prefer them.

Chameleon - a spatially aware palmtop

Fitzmaurice's *Chameleon* [9] is a spatially-aware palmtop computer. Using locational information, Chameleon allows a user to navigate through a virtual 3D space by changing the location and orientation of the palmtop in his hand. Locational information is also used to display context sensitive information in the real world. For example, by moving Chameleon toward a specific area on a wall map, information regarding that area appears on the screen. Using locational information to detect the user's circumstances, although a very good idea, has some limitations. First, location is not always enough to identify situations. When real world objects (e.g. books) move, the system can no longer keep up. Secondly, detecting the palmtop's own position is a difficult problem. The Polhemus sensor used with Chameleon has a very limited sensing range (typically 1-2 meters) and is sensitive to interference from other magnetic devices. Relying on this technology limits the user's activity to very restricted areas.

FUTURE DIRECTIONS

Situation Sensing Technologies

We are currently just using a color-code system and a CCD camera to read the code, to investigate the potential of augmented interaction. This very basic color-code system is, however, unrealistic for large scale applications, because the number of detectable IDs is quite limited. We plan to attach a line-sensor to NaviCam and use a real barcode system. This would make the system more practical.

Situation sensing methods are not limited to barcode systems. We should be able to apply a wide range of techniques to enhance the usefulness of the system.

Several, so-called next generation barcode systems have already been developed. Among them, the most appealing technology for our purposes would seem to be the *Supertag* technology invented by CSIR in South Africa [11]. Supertag is a wireless electronic label system that uses a battery less passive IC chip as an ID tag. The ID sensor is comprised of a radio frequency transmitter and a receiver. It scans hundreds of nearby tags simultaneously without contact. Such wireless ID technologies should greatly improve the usefulness of augmented interaction.

For location-detection, we could employ the global positioning system (GPS) which is already in wide use as a key-component of car navigation systems. The personal handy phone system (PHS) is another possibility. PHS is a micro-cellular wireless telephone system which will come into oper-

ation in Japan in the summer of 1995. By sensing which cell the user is in, the system can know where the user is located.

A more long-range vision would be to incorporate various kinds of vision techniques into the system. For example, if a user tapped a finger on an object appearing on the display, the system would try to detect what the user is pointing to by applying pattern matching techniques.

Obviously, combining several information sources (such as location, real world IDs, time, and vision) should increase the reliability and accuracy of situation detection, although the inherent problems are not trivial. This will also be another future direction for our research.

Inferring the user's intention from the situation

Recognized situations are still only a clue to user's intentions. Even when the system knows where the user is in and at which object the user is looking, it is not a trivial problem to infer what the user wants to know. This issue is closely related to the design of agent-based user interfaces. How do we design an agent that behaves as we would want? This is a very large open-question and we do not have immediate answer to this. It may be possible to employ various kinds of intelligent user interface technologies such as those discussed in [10].

CONCLUSION

In this paper, we proposed a simple but effective method to realize computer augmented environments. The proposed augmented interaction style focuses on human-real world interaction and not just human-computer interaction. It is designed for the highly portable and personal computers of the future, and concentrates on reducing the complexity of computer operation by accepting real world situations as implicit input. We also reported on our prototype system called Navi-Cam, which is an ID-aware palmtop system, and described some applications to show the possibilities of the proposed interaction style.

ACKNOWLEDGMENTS

We would like to thank Mario Tokoro for supporting our work. We would also like to thank Satoshi Matsuoka, Shigemitsu Oozahata and members of Sony CSL for their encouragement and helpful discussions. Special thanks also go to Yasuaki Honda for Figure 7 and assisting video production, and to Tatsuo Nagamatsu for information on the video capturing hardware.

REFERENCES

1. Ronald Azuma and Gary Bishop. Improving static and dynamic registration in an optical see-through HMD. In *Proceedings of SIGGRAPH '94*, pp. 197–204, July 1994.
2. Michael Bajura, Henry Fuchs, and Ryutarou Ohbuchi. Merging virtual objects with the real world: Seeing ultrasound imagery within the patient. *Computer Graphics*, Vol. 26, No. 2, pp. 203–210, 1992.
3. Michael Bajura and Ulrich Neumann. Dynamic registration correction in augmented-reality systems. In *Virtual Reality Annual International Symposium (VRAIS) '95*, pp. 189–196, 1995.

4. R. A. Bolt. Put-That-There: voice and gesture at the graphics interface. *ACM SIGGRAPH Comput. Graph.*, Vol. 14, No. 3, pp. 262–270, 1980.
5. Steven Feiner, Blair MacIntyre, Marcus Haupt, and Eliot Solomon. Windows on the world: 2D windows for 3D augmented reality. In *Proceedings of UIST'93, ACM Symposium on User Interface Software and Technology*, pp. 145–155, November 1993.
6. Steven Feiner, Blair MacIntyre, and Doree Seligmann. Annotating the real world with knowledge-based graphics on a see-through head-mounted display. In *Proceedings of Graphics Interface '92*, pp. 78–85, May 1992.
7. Steven Feiner, Blair MacIntyre, and Doree Seligmann. Knowledge-based augmented reality. *Communication of the ACM*, Vol. 36, No. 7, pp. 52–62, August 1993.
8. 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.
9. George W. Fitzmaurice. Situated information spaces and spatially aware palmtop computers. *Communication of the ACM*, Vol. 36, No. 7, pp. 38–49, July 1993.
10. Wayne D. Gray, William E. Hefley, and Dianne Murray, editors. *Proceedings of the 1993 International Workshop on Intelligent User Interfaces*. ACM press, 1993.
11. Peter Hawkes. Supertag - reading multiple devices in a field using a packet data communications protocol. In *CardTech/SecurTech '95*, April 1995.
12. Hiroshi Ishii. TeamWorkStation: towards a seamless shared workspace. In *Proceedings of CSCW '90*, pp. 13–26, 1992.
13. Katashi Nagao and Jun Rekimoto. Ubiquitous Talker: Spoken language interaction with real world objects. In *Proc. of IJCAI-95*, 1995.
14. Ken Sakamura. The objectives of the TRON project. In *TRON Project 1987: Open-Architecture Computer Systems*, pp. 3–16, Tokyo, Japan, 1987.
15. Ivan Sutherland. A head-mounted three dimensional display. In *Proceedings of FJCC 1968*, pp. 757–764, 1968.
16. Mark Weiser. The computer for the twenty-first century. *Scientific American*, 1991.
17. Pierre Wellner. Interacting with paper on the DigitalDesk. *Communication of the ACM*, Vol. 36, No. 7, pp. 87–96, August 1993.
18. Pierre Wellner, Wendy Mackay, and Rich Gold. Computer augmented environments: Back to the real world. *Communication of the ACM*, Vol. 36, No. 7, August 1993.

Virtual Object Manipulation on a Table-Top AR Environment

H. Kato¹, M. Billinghurst², I. Poupyrev³, K. Imamoto¹, K. Tachibana¹

¹Faculty of Information Sciences,
Hiroshima City University
3-4-1, Ozuka-higashi, Asaminami-ku,
Hiroshima, 731-3194 JAPAN
kato@sys.im.hiroshima-cu.ac.jp

²HIT Laboratory,
University of Washington
Box 352-142, Seattle,
WA 98195, USA
grof@hitl.washington.edu

³ATR 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 a different 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 [Schmalsteig 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 disconnection 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 a 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 is 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 of 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).

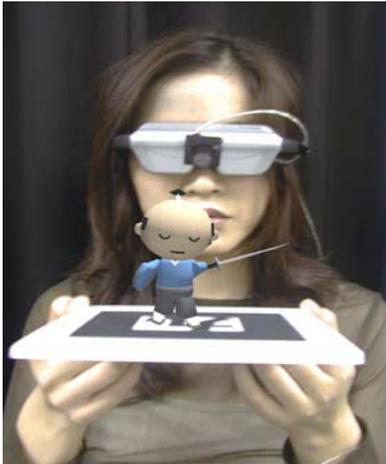


Fig. 2: A Virtual Object on a Card

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.

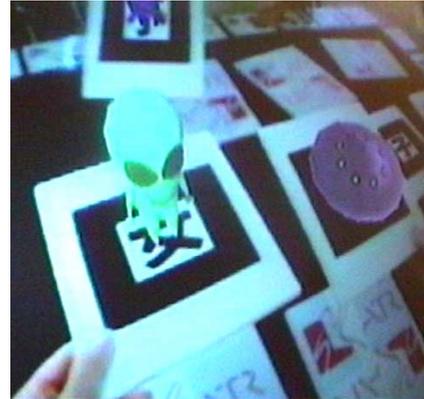


Fig. 3a: Two Matching Objects Being Brought Together

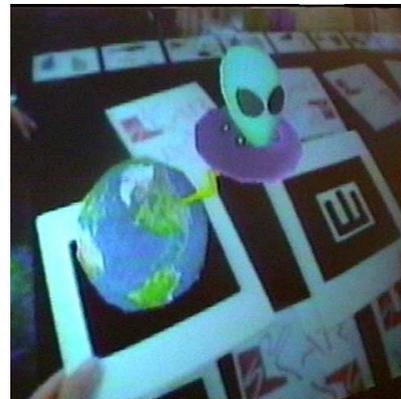


Fig. 3b: The Virtual Object Interaction

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

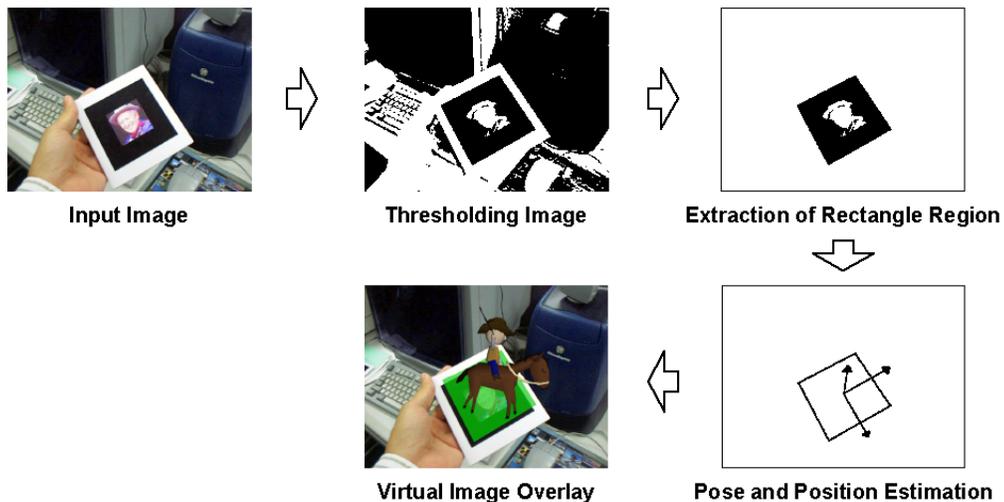


Figure 4: The Vision-Based AR Tracking Process

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 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 Coordinates 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.

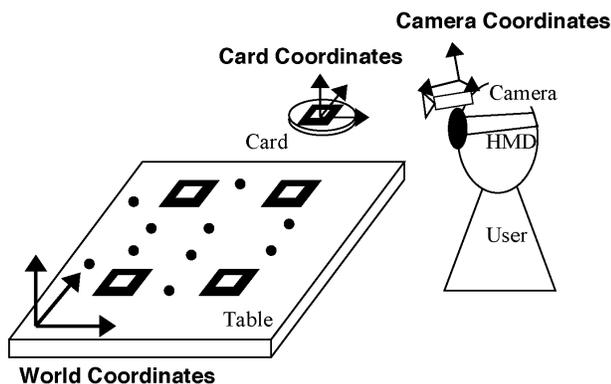


Figure 5 Table-top Configuration.

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 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 squares 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.

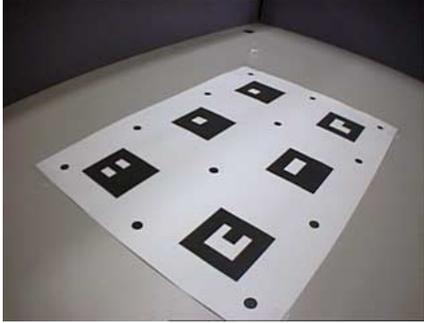


Figure 6 An Example of Fiducials.

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 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 is moved 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.

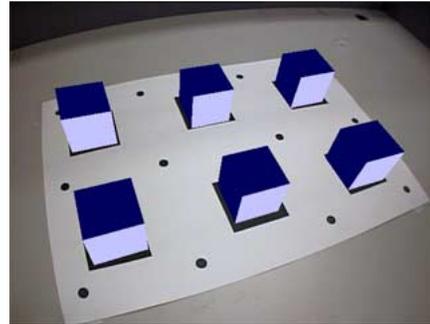


Figure 7a: Virtual Objects on Multiple Markers

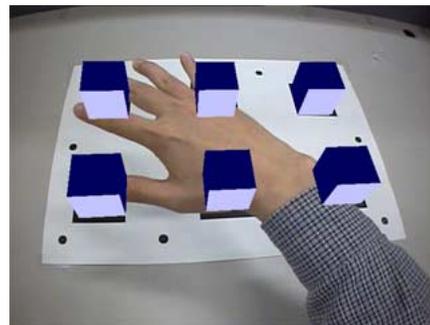


Figure 7b: Markers Covered by a Hand

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 that 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 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.

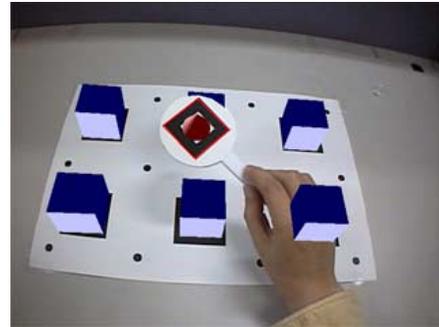


Figure 8a: correct overlay of a physical object

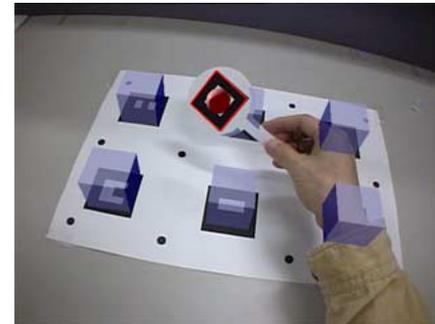


Figure 8b: transparent virtual objects

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 is a class 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 [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 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 use 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 used 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.

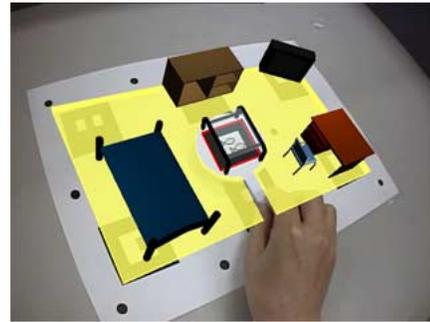


Figure 9 A Prototype of an Interior Design Application

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. 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

- [Billinghurst 99] Billinghurst, 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 98] 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.

The *Studierstube* Augmented Reality Project

Dieter Schmalstieg
dieter@cg.tuwien.ac.at
Vienna University of
Technology, Austria

Anton Fuhrmann
VRVis Research Center for
Virtual Reality and Visualization,
Vienna, Austria*

Gerd Hesina
Vienna University of
Technology, Austria

Zsolt Szalavári
Vienna University of
Technology, Austria

L. Miguel Encarnação
Fraunhofer CRCG, Inc.,
Providence, Rhode Island, U.S.

Michael Gervautz
Imagination GmbH, Vienna,
Austria*

Werner Purgathofer
Vienna University of
Technology, Austria

* Work done while at Vienna University of
Technology

Abstract

This paper describes *Studierstube*, an augmented reality system developed over the past four years at Vienna University of Technology, Austria, in extensive collaboration with Fraunhofer CRCG, Inc. in Providence, Rhode Island, U.S. Our starting point for developing the *Studierstube* system was the belief that augmented reality, the less obtrusive cousin of virtual reality, has a better chance of becoming a viable user interface for applications requiring manipulation of complex three-dimensional information as a daily routine. In essence, we are searching for a 3D user interface metaphor as powerful as the desktop metaphor for 2D. At the heart of the *Studierstube* system, collaborative augmented reality is used to embed computer-generated images into the real work environment. In the first part of this paper, we review the user interface of the initial *Studierstube* system, in particular the implementation of collaborative augmented reality, and the Personal Interaction Panel, a two-handed interface for interaction with the system. In the second part, an extended *Studierstube* system based on a heterogeneous distributed architecture is presented. This system allows the user to combine multiple approaches--augmented reality, projection displays, ubiquitous computing--to the interface as needed. The environment is controlled by the Personal Interaction Panel, a two-handed pen-and-pad interface, which has versatile uses for interacting with the virtual environment. *Studierstube* also borrows elements from the desktop, such as multi-tasking and multi-windowing. The resulting software architecture resembles in some ways what could be called an "augmented reality operating system." The presentation is complemented by selected application examples.

1. Introduction

Studierstube is the German term for the “study room” where Goethe’s famous character, Faust, tries to acquire knowledge and enlightenment (Goethe, 1808). We chose this term as the working title for our efforts to develop 3D user interfaces for future work environments. Most virtual reality systems of today are tailored to the needs of a single, very specific application that is highly specialized for that purpose. In contrast, the *Studierstube* project tries to address the question of how to use three-dimensional interactive media in a general work environment, where a variety of tasks are carried out simultaneously. In essence, we are searching for a 3D user interface metaphor as powerful as the desktop metaphor for 2D.

Our starting point for developing *Studierstube* was the belief that augmented reality (AR), the less obtrusive cousin of virtual reality (VR), has a better chance than VR of becoming a viable user interface for applications requiring information manipulation as a daily routine. Today’s information workers are required to carry out a large variety of tasks, but communication between human co-workers has an equally significant role. Consequently, *Studierstube* tries to support productivity, typically associated with the desktop metaphor, as well as collaboration, typically associated with computer supported cooperative work applications. To fulfill these needs, the framework therefore has taken on many functions of a conventional operating system in addition to being a graphical application.

At the heart of the *Studierstube* system, collaborative AR is used to embed computer-generated images into the real work environment. AR uses display technologies such as see-through head-mounted displays (HMDs) or projection screens to combine computer graphics with a user’s view of the real world. By allowing

multiple users to share the same virtual environment, computer supported cooperative work in three dimensions is enabled.

This paper gives an overview of the various avenues of research that were investigated in the course of the last four years, and how they relate to each other. The intent of this paper is to provide a summary of this rather extensive project as well as an introduction to the approach of blending augmented reality with elements from other user interface paradigms to create a new design for a convincing 3D work environment. In the first part of this paper, we review the core user interface technologies of the initial *Studierstube* work, in particular the implementation of collaborative augmented reality, and the Personal Interaction Panel, a two handed-interface for interaction with the system.

In the second part, we present an extended collaborative 3D interface that unites aspects of multiple user interface paradigms: augmented reality, ubiquitous computing, and the desktop metaphor. In the third part, we illustrate our work by reviewing some selected experimental applications that were built using *Studierstube*. Finally, we discuss how *Studierstube* is related to previous work, and draw conclusions.

2. Interaction in augmented reality

The initial *Studierstube* system as described in (Schmalstieg et al., 1996) and (Szalavári et al., 1998a) was among the first collaborative augmented reality systems to allow multiple users to gather in a room and experience a shared virtual space that can be populated with three-dimensional data. Head-tracked HMDs allow each user to choose an individual viewpoint while retaining full stereoscopic graphics. This is achieved by rendering the same virtual scene for every user’s viewpoint (or more precisely, for every user’s eyes), while taking the users’ tracked head positions into account.

Collaborators may have different preferences concerning the chosen visual representation of the data, or they may be interested in different aspects. It is also possible to render customized views of the virtual scene for every user that differ in aspects other than the viewpoint (for example, individual highlighting or annotations). At the same time, co-presence of users in the same room allows natural interaction (talking, gesturing etc.) during a discussion. The combination of real world experience with the visualization of virtual scenes yields a powerful tool for collaboration (Figure 1).



Figure 1: Two collaborators wearing see-through displays are examining a flow visualization data set

2.1 The Personal Interaction Panel

The Personal Interaction Panel (PIP) is a two-handed interface used to control *Studierstube* applications (Szalavári & Gervautz, 1997). It is composed of two lightweight hand-held props, a pen and a panel, both equipped with magnetic trackers. Via the see-through HMD, the props are augmented with computer generated images, thus instantly turning them into application-defined interaction tools similar in spirit to the virtual tricorder of Wloka & Greenfield (1995), only

using two hands rather than one. The pen and panel are the primary interaction devices.

The props' familiar shapes, the fact that a user can still see his or her own hands, and the passive tactile feedback experienced when the pen touches the panel make the device convenient and easy to use. Proprioception (Mine et al., 1997) is readily exploited by the fact that users quickly learn how to handle the props and can remember their positions and shapes. A further advantage is that users rarely complain about fatigue as they can easily lower their arms and look down on the props.

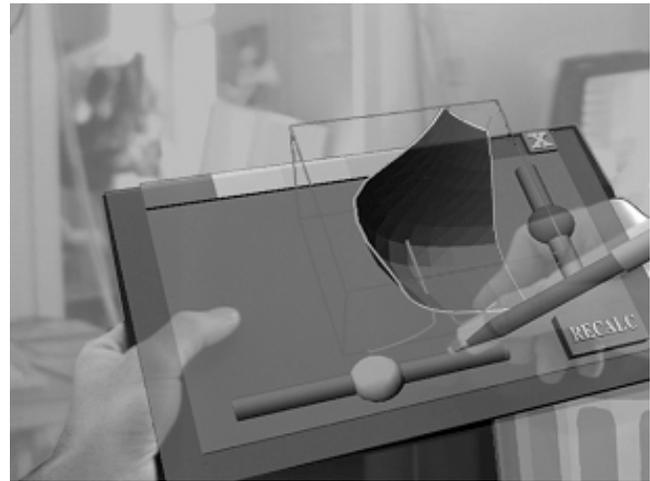


Figure 2: The Personal Interaction Panel allows two-handed interaction with 2D and 3D widgets in augmented reality

The asymmetric two-handed interaction exploits Guiard's observations (1987) that humans often use the non-dominant hand (holding the panel) to provide a frame of reference for the fine-grained manipulations carried out with the dominant hand (holding the pen). Many of the interaction styles we have designed take advantage of this fact.

However, the panel not only provides a frame of reference, but also a natural embedding of 2D in 3D (Figure 2). Many of the artifacts we encounter in real life, such as TV remote controls or button panels on household items such as microwave ovens, are essentially two-

dimensional. The PIP approach with its tactile feedback on the panel's surface resembles those real world artifacts better than naïve VR approaches such as flying menus. Consequently, the PIP provides a way to transpose many useful widgets and interaction styles from the desktop metaphor into augmented reality. Such "2.5D" widgets such as buttons, sliders or dials provide the bread-and-butter of interaction.

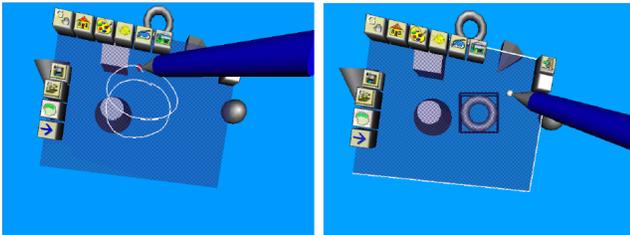


Figure 3: A gesture is used to create a torus in CADesk

However, the PIP's direct and expressive interaction language has much more to offer:

- **Object manipulation:** The pen is used as a six-degree-of-freedom pointer for object manipulation in three dimensions. Objects can either be manipulated directly in the virtual space, on the panel, or in any combination of the two. A user can instantly establish such combinations by overlaying the fixed-world frame of reference with the frame of reference defined by the panel, for example, by dragging and dropping objects from a palette to the virtual scene.
- **Gestural interaction:** Perhaps the most fundamental function of a pen and panel is gesturing, i.e., writing and drawing. As noted by (Poupyrev et al., 1998), using the panel as a surface for the gestures is an efficient mode of input in virtual environments, and even more so in AR where a user can see his or her hands while gesturing. Delimiting the area for gestures on the panel's surface allows simultaneous symbolic input and direct object manipulation. Figure 3 shows CADesk (Encarnação et al., 1999a), a solid modeling tool that has been enhanced with gesture-

based interaction using the *Studierstube* framework (Encarnação et al., 1999b).



Figure 4: The panel is used to position a clipping plane that cuts away a portion from the volumetric scan of a human skull

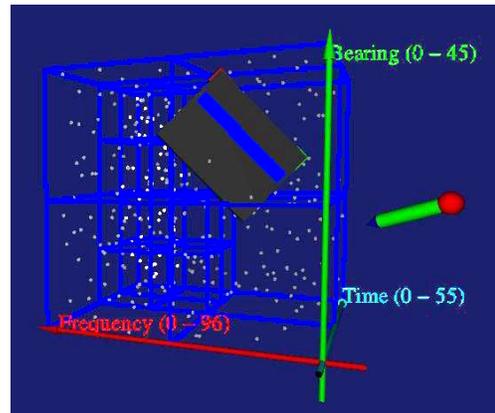


Figure 5: The panel is swept through an aggregation of particle data. During the sweep, a filter is applied to the underlying raw data, which produces aural feedback that can assist the user in detecting structures in the data sets that are not visible to the human eye.

- **Surface tool:** The panel, a two-dimensional physical shape that extends in three-dimensional space, can be interpreted as a hand-held plane or planar artifact. It can be used as a screen showing still images, animations, flat user interfaces (compare Angus & Sowizral, 1995), or live images taken from the real or virtual environment (like the screen of a digital camcorder). For example, in the MediDesk application (Wohlfahrter et al., 2000), the panel can be used to slice a volumetric model to obtain "X-ray plates" (Figure 4). Map-type tools such as worlds-in-miniature (Pausch et al., 1995) can use the panel as a ground plane. The panel

can also be used to apply filters to the data samples penetrated when sweeping the panel through a data set (Encarnaç o et al., 2000). Such filters can produce new visual representations of the underlying data sets or other kinds of feedback, such as sonification (Figure 5).

2.2 Privacy in Augmented Reality

The *personal* in Personal Interaction Panel was chosen to emphasize how its use allows users to leverage the advantages of collaborative augmented reality: Holding and manipulating the PIP puts a user in control of the application. If only one PIP is used, contention for control is resolved using social protocols such as passing on the PIP. In contrast, giving each user a separate PIP allows concurrent work. Although using multiple PIPs requires the system software to resolve the resulting consistency issues, users can freely interact with one or multiple data sets, because every user gets a separate set of controls on his or her PIP. Fuhrmann & Schmalstieg (1999) describe how interface elements can, but need not be shared by users or application instances.

The concept of personal interaction in collaborative environments is tied to the issue of privacy – users do not necessarily desire all their data to be public (Butz et al., 1998). Fortunately, a display architecture that supports independent per-user displays such as ours can be configured to use subjective views (Smith & Mariani, 1997) with per-user variations to a common scene graph. One user may display additional information that is not visible for the user’s collaborators, for example if the additional information is confusing or distracting for other users, or if privacy is desired (consider highlighting or private annotations). We found the PIP to be a natural tool for guarding such private information: For privacy, a user can make information on the panel invisible to others. This

idea was explored in (Szalav ari et al., 1998b) for collaborative games to prevent users from cheating (Figure 6).

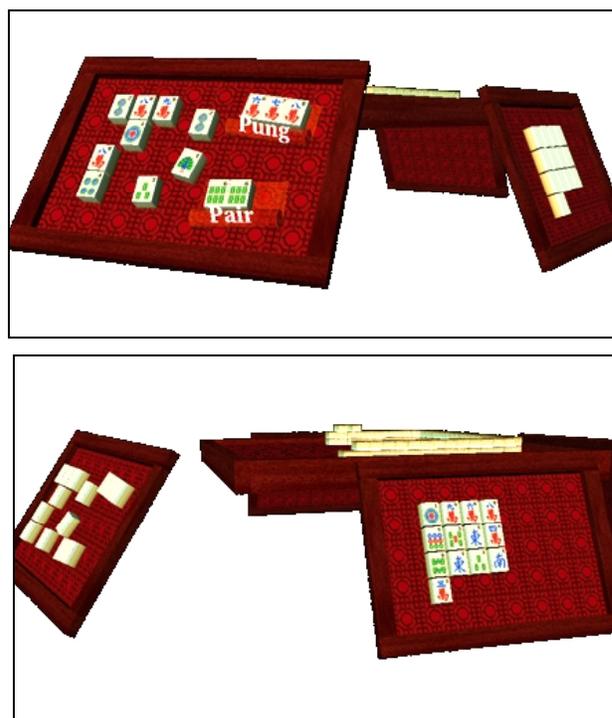


Figure 6: Personal displays secure privacy when playing Mahjongg – the left player (top view) cannot see his opponent’s tile labels and vice versa (bottom view)

2.3 Augmented Reality for the Virtual Table platform

Normally, AR is associated with see-through or video-based HMDs. Unlike HMDs, large stereo back-projection screens viewed with shutter glasses, such as used in CAVE (Cruz-Neira et al., 1993), wall, or workbench (Kr uger et al., 1995) setups, offer significantly better viewing quality, but cannot produce augmentation, as opaque physical objects will always occlude the back projection¹. To overcome this restriction, we developed a setup that achieves a kind of inverse augmented reality,

¹ Note that this discussion does not consider front projection, which is capable of producing so-called spatially augmented reality, but suffers from a different set of technical complexities.

or *augmented VR*, for the Virtual Table (VT), a workbench-like device, through the use of transparent pen and panel props made from Plexiglas (Schmalstieg et al., 1999).

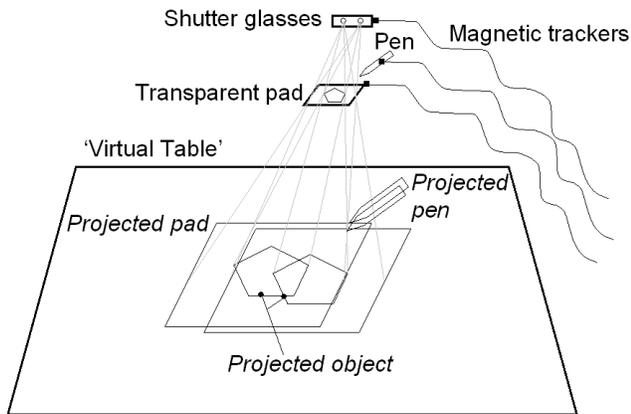


Figure 7: The Personal Interaction Panel combines tactile feedback from physical props with overlaid graphics to form a two-handed general-purpose interaction tool for the Virtual Table.

Using the information from the trackers mounted to shutter glasses and props, the workstation computes stereoscopic off-axis projection images that are perspectively correct for the user's head position. This property is essential for the use of AR as well as augmented VR, since the physical props and their virtual counterparts have to appear aligned in 3D (Figure 7). Additional users with shutter glasses can share the view with the leading user, but they experience some level of perspective distortion. Also the virtual panel will not coincide with its physical counterpart.

The material for the pen and pad was selected for minimal reflectivity, so that with dimmed lights – the usual setup for working with the VT – the props become almost invisible. While they retain their tactile property, in the user's perception they are replaced by the graphics from the VT (Figure 8).

Our observations and informal user studies indicate that virtual objects can even appear floating above the Plexiglas surface, and that

conflicting depth cues resulting from such scenarios are not perceived as disturbing. Minor conflicts occur only if virtual objects protrude from the outline of the prop as seen by the user because of the depth discontinuity. The most severe problem is occlusion from the user's hands. Graphical elements on the pad are placed in a way so that such occlusions are minimized, but they can never be completely avoided.

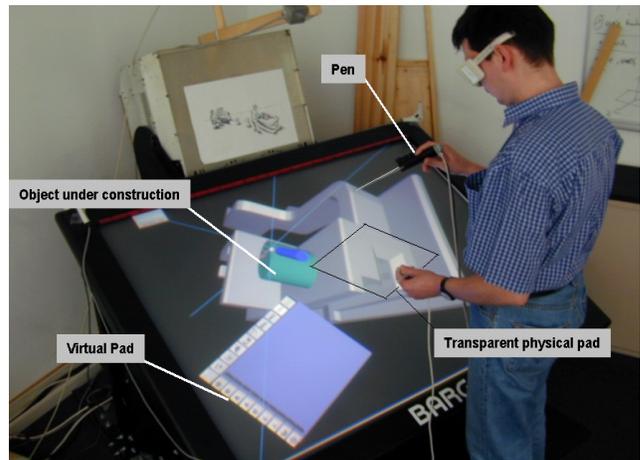


Figure 8: Transparent pen and pad for the Virtual Table are almost invisible and replaced by computer graphics in the user's perception (Stork & de Amicis, 2000)

Using the transparent props, the *Studierstube* software was ported to the VT platform. Applications could now be authored once and displayed on different platforms. One lesson we learned in the process was that the format and properties of the display strongly influence application design, much like a movie converted from Cinemascope to TV must be edited for content.

It was only after a working prototype of the VT setup was finished that we realized that a *transparent* panel affords new interaction styles because the user can see *through* it:

- **Through-the-plane tools:** The panel is interpreted as a two-dimensional frame defining a frustum-shaped volume. A single object or set of objects contained in that volume instantly becomes subject to further

manipulation – either by offering context sensitive tools such as widgets placed at the panel’s border, or by 2D gestural interaction on the panel’s surface. For example, Figure 9 shows the application of a „lasso“ tool for object selection.

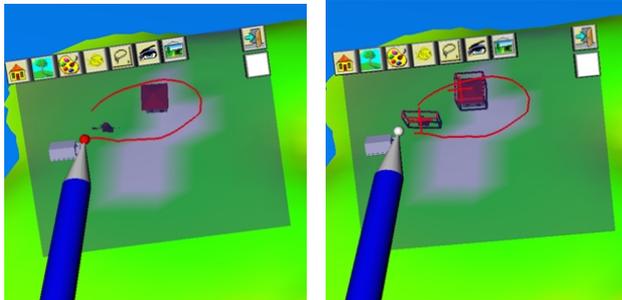


Figure 9: The lasso tool allows users to select objects in 3D by sweeping an outline in 2D on the pad. All objects whose 2D projection from the current viewpoint is contained in the outline are selected.

- **Through-the-window tools:** The transparent panel is interpreted as a window into a different or modified virtual environment. This idea includes 3D *magic lenses* (Viega et al., 1996) such as X-ray lenses (Figure 10), that are essentially modified versions of the main scene, but also SEAMS (Schmalstieg & Schafler, 1998), which are portals to different scenes or different portions of the same scene. A recent extension to the window tools is proposed in (Stoev et al., 2000): The panel acts as a lens into a separate locale of the virtual environment, the pen is used to move the scene underneath.

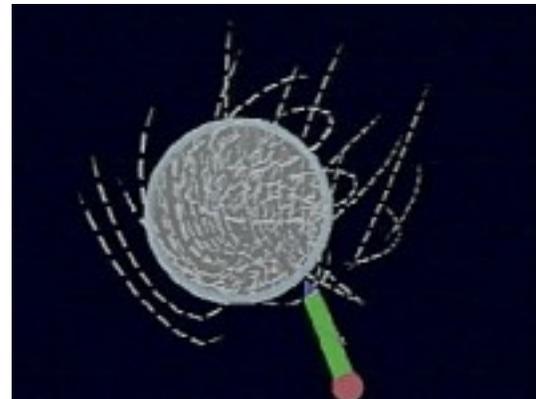


Figure 10: Different applications of through-the-window tools: (top) X-ray lens, (middle) focus lens that locally increases density of streamlines in a flow visualization, (bottom) portal to a different version of a scene

3. Convergence of user interface metaphors

During the work on the original *Studierstube* architecture, we rapidly discovered new

promising avenues of research, which could not be investigated using the initial limited design. From about 1998 on, we therefore concentrated our efforts at re-engineering and extending the initial solutions to construct a second-generation platform building on what we had learned. The support for the VT platform, as detailed in the last section, was the first outcome of this work.

It gradually became clear that augmented reality – even in a collaborative flavor – was not sufficient to address all the user interface requirements for the next generation 3D work environment we had in mind. We needed to mix and match elements from different user interface metaphors. A vision of converging different user interface paradigms evolved (Figure 11). In particular, we wanted to converge AR with elements from *ubiquitous computing* and the *desktop metaphor*.

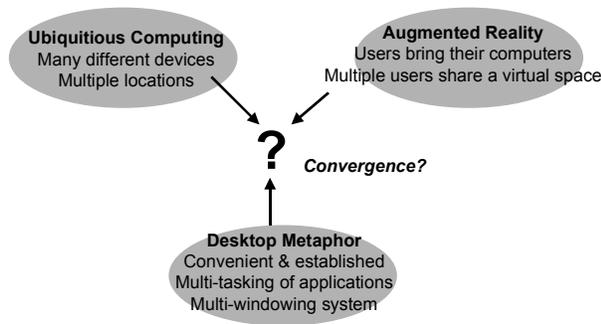


Figure 11: The latest *Studierstube* platform combines the best elements from augmented reality, ubiquitous computing, and the desktop metaphor

In contrast to AR, which is characterized by users carrying computing and display tools to augment their environment, ubiquitous computing (Weiser, 1990) denotes the idea of embedding many commodity computing devices into the environment, thus making continuous access to networked resources a reality. The VT platform, although hardly a commodity, is an instance of such a situated device. Yet there are other devices such as personal digital assistants

(PDAs) that blur the boundaries between AR and ubiquitous computing. We are interested in exploring possible combinations of a multitude of simultaneously or alternatively employed displays, input, and computing infrastructures.

While new paradigms such as AR and ubiquitous computing enable radical redesign of human-computer interaction, it is also very useful to transpose knowledge from established paradigms, in particular from the desktop, into new interaction environments. Two-dimensional widgets are not the only element of the desktop metaphor that we consider useful in a 3D work environment. Desktop users have long grown accustomed to multi-tasking of applications that complement each other in function. In contrast, many VR software toolkits allow the development of multiple applications for the same execution environment using an abstract application programmer’s interface (API); however, the execution environment usually cannot run multiple applications concurrently. Another convenient feature of desktop applications is that many of them support a multiple document interface (MDI), i.e. working with multiple documents or data sets simultaneously, allowing comparison and exchange of data among documents. The use of 2D windows associated with documents allows convenient arrangement of multiple documents according to a user’s preferences. While these properties are established in the desktop world, they are not exclusive to it and indeed useful to enhance productivity in a 3D work environment as well.

The latest version of the *Studierstube* software framework explores how to transpose these properties into a virtual environment (Schmalstieg et al., 2000). The design is built on three key elements: users, contexts, and locales.

3.1 Users

Support for multiple collaborating users is a fundamental property of the *Studierstube* architecture. While we are most interested in computer-supported face-to-face collaboration, this definition also encompasses remote collaboration. Collaboration of multiple users implies that the system will typically incorporate multiple host computers – one per user. However, *Studierstube* also allows multiple users to interact with a single host (e.g. via a large screen or a multi-headed display), and a single user to interact with multiple computers at once (by simultaneous use of multiple displays). This design is realized as a distributed system composed of different computing, input (PIP) and output (display) devices that can be operated simultaneously.

3.2 Contexts

The building blocks for organizing information in *Studierstube* are called *contexts*. A context encloses the data itself, the data's representation and an application that operates on the data. It therefore roughly corresponds to an object-oriented implementation of a document in a conventional desktop system. Users only interact within those contexts, so the notion of an application is completely hidden from the user. In particular, users never have to “start” an application; they simply open a context of a specific type. Conceptually, applications are always “on” (Kato et al., 2000).

In a desktop system, the data representation of a document is typically a single 2D window. Analogously, in our three-dimensional user interface, a context's representation is defined as a three-dimensional structure contained in a box-shaped volume – a 3D-window (Figure 12). Note that unlike its 2D counterpart, a context can be shared by any group of users.

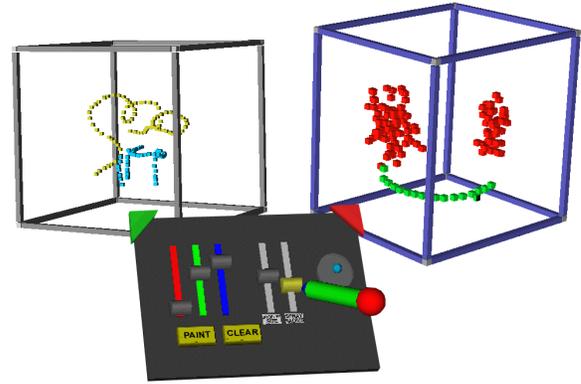


Figure 12: Multiple document interface in 3D – the right window has the user's focus – indicated by the dark window frame – and can be manipulated with the control elements on the PIP.

Every context is an instance of a particular application type. Contexts of different types can exist concurrently, which results in multi-tasking of multiple applications. Moreover, *Studierstube* also allows multiple contexts of the same type, thereby implementing an MDI. Multiple contexts of the same type are aware of each other and can share features and data. For example, consider the miniature stages of the Storyboarding application (section 8), which share the “slide sorter” view.

3.3 Locales

Locales correspond to coordinate systems in the virtual environment. They usually coincide with physical places, such as a lab or conference room or part of a room, but they can also be portable and linked to a user's position or used arbitrarily—even overlapping locales in the same physical space are allowed and used. By convention, every display used in a *Studierstube* environment shows the content of exactly one locale, but one locale can be assigned to multiple displays. Every context can—but need not—be replicated in every locale, i.e. it can appear, at most, once in every locale. All replicas of a particular context are kept synchronized by

Studierstube's distribution mechanism (section 6).

3.4 Context vs. locale

At first glance, it may not be obvious why a separation of contexts and locales is necessary. For example, the EMMIE system (Butz et al., 1999) envelops users and computers in a single environment called "ether," which is populated by graphical data items. An item's locale also defines its context and vice versa. All displays share the same physical locale. While this approach is simple to understand and easy to implement, the interaction design does not scale well with the number of data items and users: As the number of data items increases, it becomes increasingly difficult to arrange them so that all users have convenient access to all data items that they are interested in. Data items may be occluded or out of reach for convenient interaction. Even a fully untethered setup of displays and devices may be inconvenient if the environment is structured in a way that forces users to walk around in order to access frequently required data. The larger the user group is, the more likely it becomes that two users that are not in close proximity will compete for a particular data item, making optimal placement difficult or impossible. Moreover, remote collaboration is ruled out by the single locale approach, as the position of a particular data item will often be inaccessible to a remote user.

In contrast, *Studierstube* separates contexts and locales for increased flexibility. Every display uses a separate locale, i.e., a scene with an independent coordinate system. A context is placed in a locale by assigning to the context's 3D-windows a particular position within the locale. This approach allows for several strategies regarding the arrangement of contexts in the relevant locales.

A strategy of making a context available exclusively in one locale is equivalent to the

single locale approach, with the exception that the locale is broken up into disjointed parts. Again, users may not be able to access desired contexts (Figure 13, top). In contrast, a strategy of replicating every context in every locale guarantees convenient access to a context, but quickly leads to display clutter (Figure 13, middle).

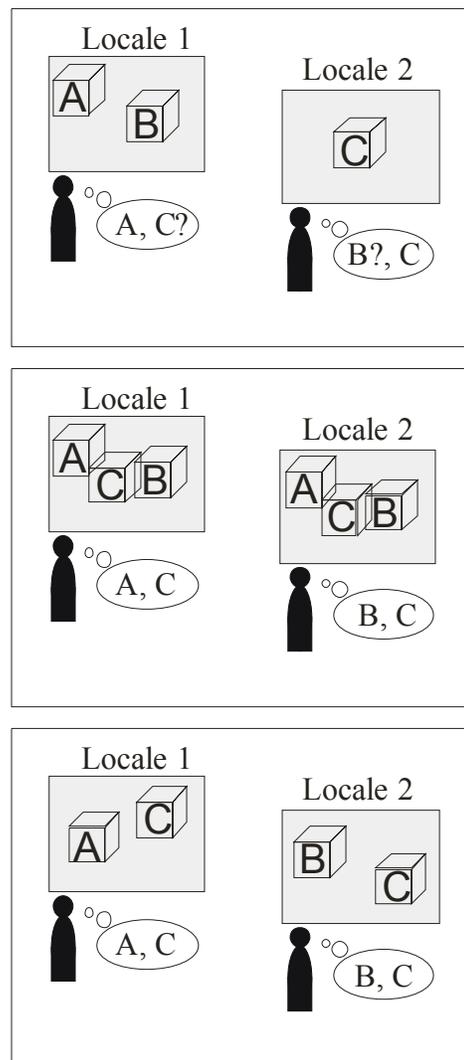


Figure 13: (top) A global arrangement of items cannot fulfill all needs. (middle) Full replication of all items leads to display clutter. (bottom) On-demand replication of items allows convenient customization of locales.

Therefore replication of a context in a given locale is optional: There may be at most one replica of a given context in a given locale. This strategy allows a user to arrange a convenient working set of contexts in his or her preferred display (Figure 13, bottom). If the displays are connected to separate hosts in a distributed system, only those hosts that replicate a context need to synchronize the context's data. If it can be assumed that working sets typically do not exceed a particular size, the system will scale well.

Yet in many situations it is desirable to share position and configuration over display boundaries. *Studierstube* thus allows locales to be shared over displays. More precisely, multiple displays can have independent points of view, but show images of an identical scene graph.

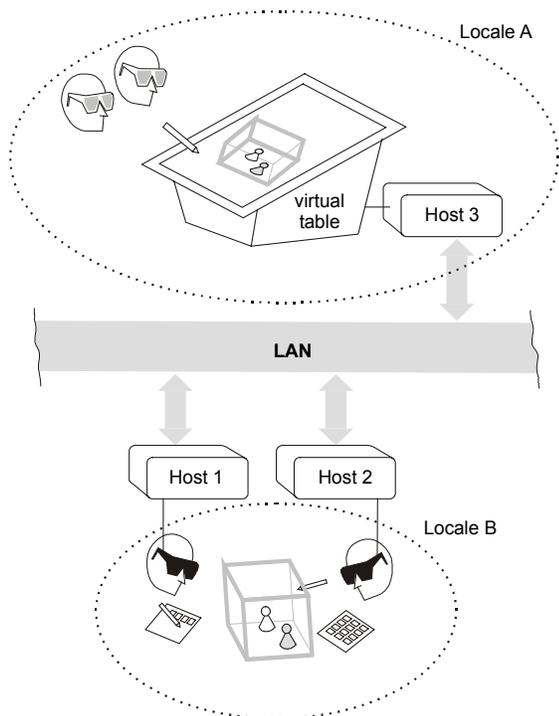


Figure 14: Multiple locales can simultaneously exist in *Studierstube*. They can be used to configure different output devices and to support remote collaboration.

This allows for collaborative augmented reality settings as introduced in section 2, but

even for more complex setups such as a large projection screen display augmented by graphics from a see-through HMD. Figure 14 shows a non-trivial example involving one context, two locales, three displays, and four users.

4. Implementation of the user interface

4.1 Software architecture

Studierstube's software development environment is realized as a collection of C++ classes built on top of the Open Inventor (OIV) toolkit (Strauss & Carey, 1992). The rich graphical environment of OIV allows rapid prototyping of new interaction styles. The file format of OIV enables convenient scripting, overcoming many of the shortcomings of compiled languages without compromising performance. At the core of OIV is an object-oriented scene graph storing both geometric information and active interaction objects. Our implementation approach has been to extend OIV as needed, while staying within OIV's strong design philosophy (Wernecke, 1994).

This has led to the development of two intertwined components: A toolkit of extensions of the OIV class hierarchy—mostly interaction widgets capable of responding to 3D events—and a runtime framework which provides the necessary environment for *Studierstube* applications to execute (Figure 15). Together these components form a well-defined application programmer's interface (API), which extends the OIV API, and also offers a convenient programming model to the application programmer (section 7).

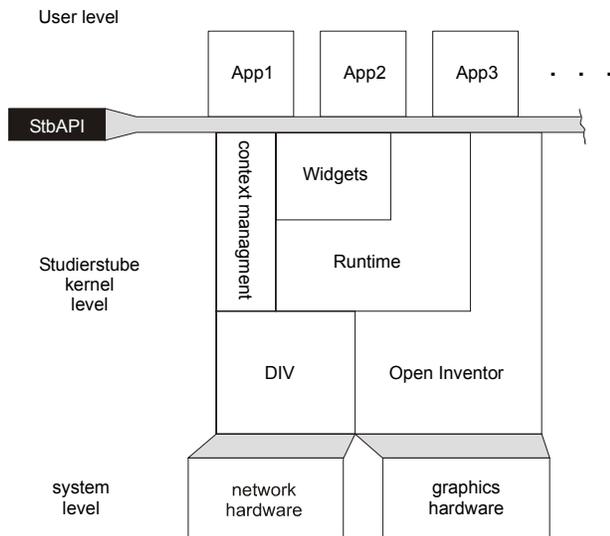


Figure 15: The *Studierstube* software is composed of an interaction toolkit and runtime system. The latter is responsible for managing context and distribution.

Applications are written and compiled as separate shared objects, and dynamically loaded into the runtime framework. A safeguard mechanism makes sure that only one instance of each application's code is loaded into the system at any time. Besides decoupling application development from system development, dynamic loading of objects also simplifies distribution, as application components can be loaded by each host whenever needed. All these features are not unique to *Studierstube*, but they are rarely found in virtual environment software.

By using this dynamic loading mechanism, *Studierstube* supports multi-tasking of *different* applications (e.g. a medical visualization and a 3D modeler) and also an MDI.

Depending on the semantics of the associated application, ownership of a context may or may not privilege a user to perform certain operations on the information (such as object deletion). Per default, users present in the same locale will share a context. Per default, a context is visible to all users and can be manipulated by any user in the locale.

4.2 Three-dimensional windows

The use of windows as an abstraction and interaction metaphor is an established convention in 2D GUIs. Its extension to three dimensions can be achieved in a straightforward manner (Tsao & Lumsden, 1997): Using a box instead of a rectangle seems to be the easiest way of preserving the well-known properties of desktop windows when migrating into a virtual environment. It supplies the user with the same means of positioning and resizing the display volume and also defines its exact boundaries.

A context is normally represented in the scene by a 3D window, although a context is allowed to span multiple windows. The 3D-window class is a container associated with a user-specified scene graph. This scene graph is normally rendered with clipping planes set to the faces of the containing box so that the content of the window does not protrude from the window's volume. Nested windows are possible, although we have found little use for them. The window is normally rendered with an associated "decoration" that visually defines the window's boundaries and allows it to be manipulated with the pen (move, resize etc). The color of the decoration also indicates whether a window is active (and hence receives 3D events from that user). Like their 2D counterparts, 3D-windows can be minimized (replaced by a three-dimensional icon on the PIP to save space in a cluttered display), and maximized (scaled to fill the whole work area). Typically, multiple contexts of the same type will maintain structurally similar windows, but this decision is at the discretion of the application programmer.

4.3 PIP sheets

Studierstube applications are controlled either via direct manipulation of the data presented in 3D-windows, or via a mixture of 2D and 3D widgets on the PIP. A set of controls on the PIP—a *PIP sheet*—is implemented as an

OIV scene graph composed primarily of *Studierstube* interaction widgets (such as buttons, etc.). However, the scene graph may also contain geometries (e. g., 2D and 3D icons) that convey the user interface state or can be used merely as decoration.

Every type of context defines a PIP sheet template, a kind of application resource. For every context and user, a separate PIP sheet is instantiated. Each interaction widget on the PIP sheet can therefore have a separate state. For example, the current paint color in an artistic spraying application can be set individually by every user for every context. However, widgets can also be shared by all users and/or all contexts. Consequently, *Studierstube*'s 3D event routing involves a kind of multiplexer between windows and users' PIP sheets.

5. Hardware support

5.1 Displays

Studierstube is intended as an application framework that allows the use of a variety of displays, including projection based devices and HMDs. There are several ways of determining camera position, creating stereo images, setting a video mode etc. After some consideration, we implemented an OIV compatible viewer with a plug-in architecture for camera control and display mode.

The following display modes are supported:

- Field sequential stereo: Images for left/right eye output in consecutive frames
- Line interleaved stereo: Images for left/right eye occupy odd/even lines in a single frame
- Dual screen: Images for left/right eye are output on two different channels
- Mono: The same image is presented to both eyes

The following camera control modes are supported:

- Tracked display: Viewpoint and display surface are moving together and are tracked (usually HMD)
- Tracker head: A user's viewpoint (head) is tracked, but the display surface is fixed (such as a workbench or wall)
- Desktop: The viewpoint is either assumed stationary, or can be manipulated with a mouse

This approach, together with a general off-axis camera implementation, allows runtime configuration of almost any available display hardware. Table 1 shows an overview of some devices that have evaluated so far.

	Tracked display	Tracked head	Desktop
Field sequential	Sony Glasstron	Virtual Table	Fishtank VR with shutter glasses
Line interleaved	i-glasses	VREX VR2210 projector	i-glasses w/o head tracking
Dual screen	i-glasses Protec	Single user dual-projector passive stereo w/head track.	Multi-user dual-projector passive stereo
Mono	i-glasses (mono)	Virtual Table (mono)	Desktop viewer

Table 1: All combinations of camera control and display modes have distinct uses.

5.2 Tracking

A software system like *Studierstube* that works in a heterogeneous distributed infrastructure and is used in several research labs with a variety of tracking devices requires an abstract tracking interface. The approach taken by most commercial software toolkits is to implement a device driver model, thereby providing an abstract interface to the tracking devices, while hiding hardware dependent code inside the supplied device drivers. While such a model is certainly superior to hard-coded device

support, we found it insufficient for our needs in various aspects:

- **Configurability:** Typical setups for tracking in virtual environments are very similar in the basic components, but differ in essential details such as the placement of tracker sources or the number and arrangement of sensors. The architecture allows the configuration of all of those parameters through simple scripting mechanisms.
- **Filtering:** There are many necessary configuration options that can be characterized as filters, i.e., modifications of the original data. Examples include geometric transformations of filter data, prediction, distortion compensation, and sensor fusion from different sources.
- **Distributed execution and decoupled simulation:** Processing of tracker data can become computationally intensive, and it should therefore be possible to distribute this work over multiple CPUs. Moreover, tracker data should be simultaneously available to multiple users in a network. This can be achieved by implementing the tracking system as a loose ensemble of communicating processes, some running as service processes on dedicated hosts that share the computational load and distribute the available data via unicast and multicast mechanisms, thereby implementing a decoupled simulation scheme (Shaw et al., 1993).
- **Extensibility:** As a research system, *Studierstube* is frequently extended with new experimental features. A modular, object-oriented architecture allows the rapid development of new features and uses them together with existing ones.

The latest version of tracking support in *Studierstube* is implemented as an object-oriented framework called OpenTracker (Reitmayr & Schmalstieg, 2000), which is available as open

source. It is based on a graph structure composed of linked nodes: source nodes deliver tracker data, sink nodes consume data for further processing (e. g. to set a viewpoint), while intermediate nodes act as filters. By adding new types of nodes, the system can easily be extended. Nodes can reside on different hosts and propagate data over a network for decoupled simulation. By using an XML (Bray et al., 2000) description of the graph, standard XML tools can be applied to author, compile, document, and script the OpenTracker architecture.

6. Distributed execution

The distribution of *Studierstube* requires that for each replica of a context, all graphical and application-specific data is locally available. In general, applications written with OIV encode all relevant information in the scene graph, so replicating the scene graph at each participating host already solves most of the problem.

6.1 Distributed shared scene graph

Toward that aim, Distributed Open Inventor (DIV) was developed (Hesina et al., 1999) as an extension—more a kind of plug-in—to OIV. The DIV toolkit extends OIV with the concept of a distributed shared scene graph, similar to distributed shared memory. From the application programmer's perspective, multiple workstations share a common scene graph. Any operation applied to a part of the shared scene graph will be reflected by the other participating hosts. All this happens to the application programmer in an almost completely transparent manner by capturing and distributing OIV's notification events.

Modifications to a scene graph can either be updates of a node's fields, i.e., attribute values, or changes to the graph's topology, such as adding or removing children. All these changes to the scene graph are picked up by an OIV sensor and reported to a DIV observer which propagates the

changes via the network to all hosts that have a replica of the context's scene graph, where the modifications are duplicated on the remote scene graph by a DIV listener (Figure 16).

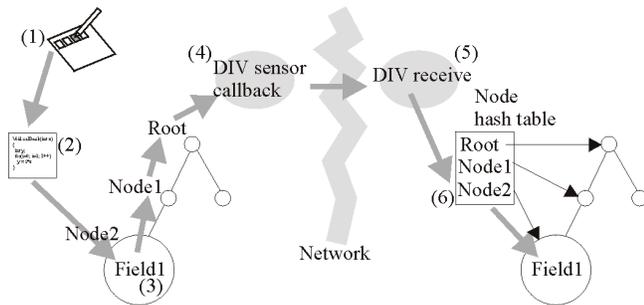


Figure 16: Example of a field update in a master-slave configuration. (1) User triggers an action by pressing a button. (2) Corresponding callback is executed and modified field1 of node2. (3) Event notification is propagated upwards in scene graph and observed by sensor. (4) Sensor transmits message to slave host. (5) Receiver picks up message and looks up corresponding node in internal hash table. (6) Slave node is modified.

On top of this master/slave mechanism for replication, several network topology schemes can be built. A simple reliable multicasting scheme based on time stamps is used to achieve consistency.

6.2 Distributed context management

A scene graph shared with DIV need not be replicated in full—only some portions can be shared, allowing local variations. In particular, every host will build its own scene graph from the set of replicated context scene graphs.

These locally varied scene graphs allow for the management of locales by resolving distributed consistency on a *per-context* basis. There exists exactly one workstation, which owns a particular context and will be responsible for processing all relevant interaction concerning the application. This host's replica is called the *master context*. All other hosts may replicate the context as a *slave context*.

The slave contexts' data and representation (window, PIP sheet etc.) stay synchronized over

the whole life span of the context for every replica.

The replication on a per-context basis provides coarse-grained parallelism. At the same time the programming model stays simple and the programmer is relieved of solving difficult concurrency issues since all relevant computation can be performed in a single address space.

The roles that contexts may assume (master or slave) affect the status of the context's application part. The application part of a master context is active and modifies context data directly according to the users' input. In contrast, a slave context's application is dormant and does not react to user input. For example, no callbacks are executed if widgets are triggered. Instead, a slave context relies on updates to be transmitted via DIV. When the application part changes the scene graph of the master context, DIV will pick up the change and propagate it to all slave contexts to keep them in sync with the master context. This process happens transparently within the application, which uses only the master context's scene graph.

Note that context replicas can swap roles (e. g., by exchanging master and slave contexts to achieve load balancing), but at any time there may only be one master copy per replicated context.

Because the low-level replication of context data is taken care of by DIV, the high-level context management protocol is fairly simple: A dedicated session manager process serves as a mediator among hosts as well as a known point of contact for newcomers. The session manager does not have a heavy workload compared to the hosts running the *Studierstube* user interface, but it maintains important directory services. It maintains a list of all active hosts and which contexts they own or subscribe to, and it determines policy issues, such as load balancing, etc.

Finally, input is managed separately by dedicated device servers (typically PCs running Linux), which also perform the necessary filtering and prediction. The tracker data is then multicast in the LAN, so it is simultaneously available to all hosts for rendering.

7. Application programmer's interface

The *Studierstube* API imposes a certain programming model on applications, which is embedded in a foundation class, from which all *Studierstube* applications are derived. By overloading certain polymorphic methods of the foundation class, a programmer can customize the behavior of the application. The structure imposed by the foundation class supports multiple contexts.

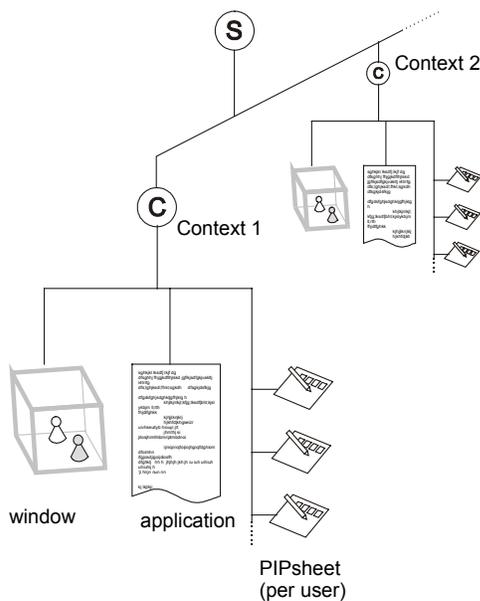


Figure 17: A context is implemented as a node in the scene graph, as are windows and PIP sheets. This allows for the organization of all relevant data in the system in a single hierarchical data structure.

Each context can be operated in both master mode (normal application processing) and slave mode (same data model, but all changes occur remotely through DIV). The key to achieving all

of this is to make the context itself a *node* in the scene graph. Such context nodes are implemented as OIV *kit* classes. Kits are special nodes that can store both fields, i.e., simple attributes, and child nodes, both of which will be considered part of the scene graph and thus implicitly be distributed by DIV. Default parts of every context are at least one 3D-window node, which is itself an OIV kit and contains the context's "client area" scene graph, and a set of PIP sheets (one for each participating user). In other words, data, representation, and application are all embedded in a single scene graph (Figure 17), which can be conveniently managed by the *Studierstube* framework.

To create a useful application with all the properties mentioned above, a programmer need only create a subclass of the foundation class and overload the 3D-window and PIP sheet creation methods to return custom scene graphs. Typically, most of the remaining application code will consist of callback methods responding to certain 3D events such as a button press or a 3D direct manipulation event. Although the programmer has the freedom to use anything that the OIV and *Studierstube* toolkits offer, any instance data is required to be stored in the derived context class as a field or node, or otherwise it will not be distributed. However, this is not a restriction in practice, as all basic data types are available in both scalar and vector formats as fields, and new types can be created should the existing ones turn out to be insufficient (a situation that has not occurred to us yet).

Note that allowing a context to operate in either master and slave mode has implications on how contexts can be distributed: It is not necessary to store all master contexts of a particular type at one host. Some master contexts may reside on one host, some on another host—in that case, there usually will be corresponding slave contexts at the respective other host, which are also instances of the same kit class, but

initialized to function as slaves. In essence, *Studierstube*'s API provides a distributed multiple document interface.

8. Applications

To evaluate the *Studierstube* platform, a number of applications were developed and are still being developed. They cover a variety of fields, for example, scientific visualization (Fuhrmann et al., 1998), CAD (Encarnação et al., 1999a), and landscape design (Schmalstieg et al., 1999). In this section, three application examples are chosen to highlight the platform's strengths: Section 8.1 discusses storyboard, a multi-user design system, section 8.2 presents MediDesk, a medical visualization tool, and section 8.3 describes Construct3D, a geometry education tool.

8.1 Storyboard design

To demonstrate the possibilities of a heterogeneous virtual environment, we chose the application scenario of *storyboard design*. This application is a prototype of a cinematic design tool. It allows multiple users to concurrently work on a storyboard for a movie or drama. Individual scenes are represented by their stage sets, which resemble *worlds in miniature* (Pausch et al., 1995).

Every scene is represented by its own context and embedded in a 3D-window. Users can manipulate the position of props in the scene as well as the number and placement of actors (represented by colored board game figures), and finally the position of the camera (Figure 18).

All contexts share an additional large *slide show* window, which shows a 2D image of the selected scene from the current camera position. By flipping through the scenes in the given sequence, the resulting slide show conveys the visual composition of the movie.

Alternatively, a user may change the slide show to a "slide sorter" view inspired by current

presentation graphics tools, where each scene is represented by a smaller 2D image, and the sequence can be rearranged by simple drag and drop operations. The slide sorter comes closest to the traditional storyboard used in cinematography. It appears on the PIP for easy manipulation as well as on the larger projection screen.

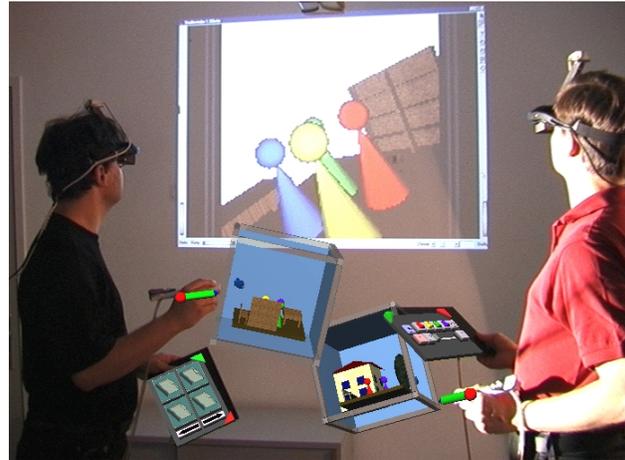


Figure 18: Storyboard application with two users and two contexts as seen from a third "virtual" user perspective, used for video documentation. In the background the video projection is visible.

The test configuration consisted of three hosts (SGI Indigo2 and O2 running IRIX, Intergraph TZ1 Wildcat running Windows NT), two users, and two locales (Figure 19). It was designed to show the convergence of multiple users (real ones as well as virtual ones), contexts, locales, 3D-windows, hosts, displays and operating systems.

The two users were wearing HMDs, both connected to the Indigo2's multi-channel output, and seeing head-tracked stereoscopic graphics. They were also fitted with a pen and panel each. The Intergraph workstation was driving an LCD video projector to generate a monoscopic image of the slide show on the projection screen (without viewpoint tracking), which complemented the presentation of the HMDs.

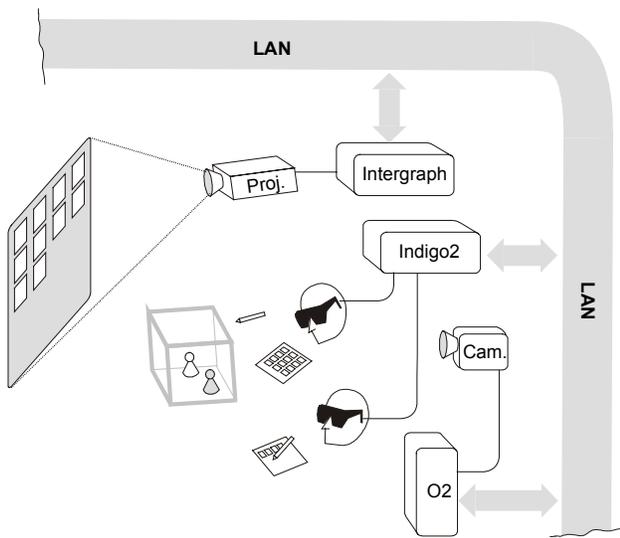


Figure 19: Heterogeneous displays—two users simultaneously see shared graphics (via their see-through HMDs) and a large screen projection.

Users were able to perform some private editing on their local contexts and then update the slide show/sorter to discuss the results. Typically, each user would work on his or her own set of scenes. However, we chose to make all contexts visible to both users so collaborative work on a single scene was also possible. The slide sorter view was shared between both users so global changes to the order of scenes in the movie were immediately recognizable.

The third host—the O2—was configured to combine the graphical output (monoscopic) from *Studierstube* with a live video texture obtained from a video camera pointed at the users and projection screen. The O2 was configured to render images for a virtual user whose position was identical with the physical camera. This feature was used to document the system on video.

The configuration demonstrates the use of overlapping locales: The first locale is shared by the two users to experience the miniature stages at the same position. This locale is also shared by the O2, which behaves like a passive observer of

the same virtual space, while a second separate locale was used for the Intergraph driving the projection screen, which could be freely repositioned without affecting the remainder of the system.

8.2 Medical visualization

MediDesk is an application for interactive volume rendering in the *Studierstube* system (Wohlfahrter et al., 2000). As the name suggests, its primary use lies in the field of medical visualization. Users can load volumetric data sets (typically CT or MRI scans), which are rendered using OpenGL Volumizer (Eckel, 1998). Volumizer allows interactive manipulation of volume data, although it requires a high-end SGI workstation for reasonable performance.



Figure 20: MediDesk allows interactive manipulation of volumetric data, such as CT scans.

As with most *Studierstube* applications, a set of buttons and sliders on the panel allows a user to control the application, such as altering transfer function parameters (Figure 20). The backside of the panel serves special purposes for volume manipulation: This allows for the design of an intuitive interface for volume rendering, a style inspired by a medical doctor's X-ray workplace.

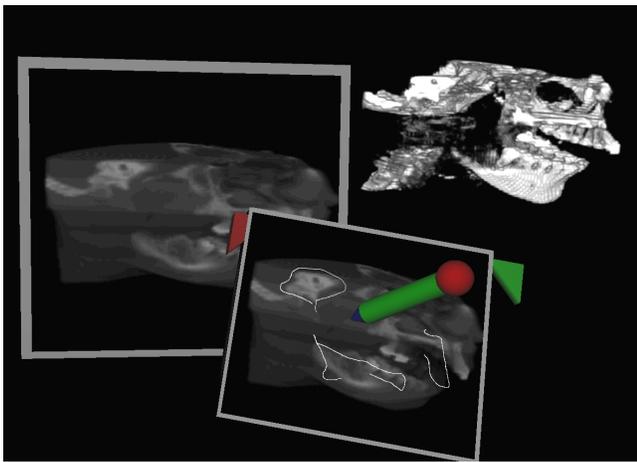


Figure 21: The lower half of the image shows the annotating of a virtual “X-ray” print taken from the volume on the upper right.

The use of two-handed interaction for manipulation of medical data has been found advantageous in the past (Goble et al., 1995). The PIP allows a similar approach: The volumetric data set can be positioned with the pen, while the panel acts as a clipping plane. The user may also freeze one or multiple clipping planes in space to inspect isolated regions of interest. Alternatively, cross-sections can be extracted from the volume with the panel and subsequently appear (as textures) on the pad, where they can be annotated with the pen as if the panel were a notepad. These virtual “X-ray” prints can be attached to a physical wall for reference (Figure 21).

8.3 Geometry education

Construct3D is a prototype application for exploring the use of collaborative augmented reality in mathematics and geometry education (Kaufmann et al., 2000). More specifically, we were interested how constructive geometry education, which still uses traditional pen-and-paper drawing methods to teach high school and college students the basics of three-dimensional space, could be implemented in *Studierstube*. It is important to note that this differs from typical computer aided design (CAD) tasks. Users trained in desktop CAD tools may have a

different background and a different set of expectations than students involved in pen and paper exercises.

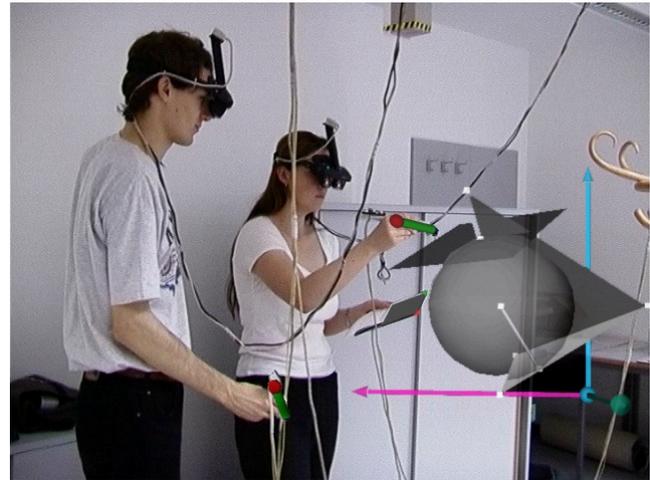


Figure 22: A tutor teaches a student how to geometrically construct 3D entities with Construct3D.

To assess the usability of a 3D tool like *Studierstube*, we found geometry education an interesting application field because is not so much concerned with the final result of the modeling, but rather with the process of construction itself and its mathematical foundation. We tried to evaluate the advantages of actually *seeing* three-dimensional objects, as opposed to calculating and constructing them using two-dimensional views. We speculated that AR would allow a student to enhance, enrich and complement the mental pictures of complex spatial problems and relationships that students form in their minds when working with three-dimensional objects. By working directly in 3D space, it may be possible to comprehend the task better and faster than with traditional methods.

We therefore aimed not at creating a professional 3D modeling package but rather at developing a simple and intuitive 3D construction tool in an immersive AR environment for educational purposes. The main goal was to keep the user interface as simple as possible to facilitate learning and efficient use. The main

areas of application of the system in mathematics and geometry education were vector analysis and descriptive geometry.

Construct3D uses the PIP to offer a palette of geometric objects (point, line, plane, box, sphere, cone and cylinder) that can be input using direct manipulation for coordinate specification (point and click). A coordinate skitter (Bier, 1986) aids accurate positioning. The modeling process is constructive in the sense that more complex primitives can be assembled from simpler ones (e. g., a plane can be defined by indicating a previously created point and line). Audio feedback guides the construction process. The use of transparency for primitives allowed users to observe necessary details, such as intersections.

With this application, an informal user study with 14 subjects was conducted. The test session consists of two parts. The first part required each participant to solve a construction example from mathematics education with the help of a tutor in Construct3D (Figure 22). The example stems from vector analysis as taught in 10th grade in Austria. For high school students, calculating the results would be lengthy and rather complex. In the second part, all subjects completed a brief survey. The survey contains an informal section about VR in general and questions about Construct3D.

In general, speculations that AR is a useful tool for geometry education were confirmed. The subjects were able to perform the task after a few minutes of initial instruction. The majority of comments regarding the AR interface were encouraging. Some questions arose about how larger groups of students could work together (we partly relate this comment to the current tethered setup that has a rather limited working volume). Some comments addressed the technical quality (such as tracking or frame rate). Most students consider AR a useful complement (but not replacement) to traditional pen and paper education. Figure 22 also shows how unplanned

uses of the environment can arise—one student spontaneously placed the printed task description on the PIP, thus “augmenting” her PIP with a physical layer of information.

9. Related work

The current architecture of *Studierstube* has absorbed many different influences and is utilizing—partially enhancing—many different ideas. The most influential areas are augmented reality, computer supported cooperative work, ubiquitous computing, and heterogeneous user interfaces. Here the discussion is limited to some of the most influential work:

The Shared Space (Billinghurst et al., 1996; Billinghurst et al., 1998b) project at University of Washington’s HITLab has—together with *Studierstube*—pioneered the use of collaborative augmented reality. Since then, HITLab has worked on many innovative applications blending AR with other components into a heterogeneous environment: Easily deployable optical tracking allows to utilize tangible objects for instant augmentation (Kato et al., 2000), for example, to build wearable augmented video conferencing spaces (Billinghurst et al., 1998a) or hybrids of AR and immersive virtual worlds.

The Computer Graphics and User Interfaces lab at Columbia University has a long reputation for augmented reality research (Feiner et al., 1993). Their EMMIE system (Butz et al., 1999) is probably the closest relative to *Studierstube*. It envelops computers and users in a collaborative “ether” populated with graphical data items provided by AR and ubiquitous computing devices such as HMDs, notebooks, PDAs, and projection walls. Communication between stationary and mobile AR users is facilitated as well (Höllner et al., 1999). Except for the locale concept, EMMIE shares many basic intentions with our research, in particular concurrent use of heterogeneous media in a collaborative work environment. Like us, (Butz et al., 1999) believe

that future user interfaces will require a broader design approach integrating multiple user interface dimensions before a successor to the desktop metaphor can emerge.

Rekimoto has developed a number of setups for multi-computer direct manipulation to bridge heterogeneous media. In (Rekimoto, 1997), a stylus is used to drag and drop data across display boundaries, while Hyperdragging (Rekimoto & Saitoh, 1999) describes a similar concept that merges multiple heterogeneous displays to create a hybrid virtual environment.

The Tangible Media Group at MIT has developed a number of heterogeneous user interfaces based on the theme of tangible (physical) objects (Ishii & Ulmer, 1997). For example, the metaDESK (Ulmer & Ishii, 1997) combines tangible objects with multiple displays, implicitly defining two views into one locale. The luminous room (Underkoffler, 1999) allows remote collaboration using embedded displays, while mediaBLOCKS (Ulmer & Ishii, 1998) are tangible containers that roughly correspond to contexts in *Studierstube*.

The Office of the Future project at UNC (Raskar et al., 1998a) is concerned with the seamless embedding of computer controlled displays into a conventional office environment. This system uses sophisticated front projection to implement spatially augmented reality (Raskar, 1998b), an interesting variety of AR.

CRYSTAL (Tsao & Lumsden, 1997) is a single-user multi-application platform. While it is agnostic in terms of display media, it pioneers the use of 3D-windows and multi-tasking of applications in virtual environments.

Finally, SPLINE (Barrus et al., 1996) is a distributed multi-user environment. From it the term “locale” is borrowed, which in SPLINE is used to describe non-overlapping places. While SPLINE is neither an AR system nor a 3D work environment (according to our use of the term), it

allows multiple users to participate in multiple activities (i.e., applications) simultaneously.

10. Conclusions and future work

Studierstube is a prototype system for building innovative user interfaces that use collaborative augmented reality. It is based on a heterogeneous distributed system based on a shared scene graph and a 3D interaction toolkit. This architecture allows for the amalgamation of multiple approaches to user interfaces as needed: augmented reality, projection displays, ubiquitous computing. The environment is controlled by a two-handed pen-and-pad interface, the Personal Interaction Panel, which has versatile uses for interacting with the virtual environment. We also borrow elements from the desktop, such as multi-tasking and multi-windowing. The resulting software architecture resembles in some ways what could be called an “augmented reality operating system.”

Research that is currently in its initial phase will investigate the possibilities of mobile collaborative augmented reality. The name *Studierstube* (“study room”) may be no longer appropriate for a portable or wearable AR platform, but a mobile 3D information platform has exciting new possibilities, such as ad-hoc networking for instant collaboration of augmented users. Our goal is to allow users to take 3D contexts “on the road” and even dock into a geographically separate environment without having to shut down live applications.

Acknowledgments

The *Studierstube* project is sponsored by the Austrian Science Fund *FWF* under contracts no. P12074-MAT and P14470-INF, by the Fraunhofer IGD/CRCG Student and Scholar Exchange Program, and the Fraunhofer IGD TRADE Virtual Table 1998-1999 Program. Special thanks to Oliver Bimber, Pedro Branco, Hannes Kaufmann, Markus Krutz, Clemens

Pecinovsky, Roman Rath, Gerhard Reitmayr, Gernot Schaufler, Rainer Splechtna, Stan Stoev, André Stork, Hermann Wurnig, and Andreas Zajic for their contributions, and to M. Eduard Gröller for his spiritual guidance.

Web information

<http://www.cg.tuwien.ac.at/research/vr/studierstube/>

References

- (Angus & Sowizral, 1995) I. Angus, H. Sowizral. Embedding the 2D Interaction Metaphor in a Real 3D Virtual Environment. Proceedings SPIE, vol. 2409, pp. 282-293, 1995.
- (Barrus et al., 1996) J. Barrus, R. Waters, R. Anderson. Locales and Beacons: Precise and Efficient Support for Large Multi-User Virtual Environments. Proc. VRAIS '96, pp. 204-213, 1996.
- (Bier, 1986) E. Bier. Skitters and Jacks: Interactive 3D Positioning Tools. Proc. 1986 ACM Workshop on Interactive 3D Graphics, Chapel Hill, NC, pp. 183-196, 1986.
- (Billinghurst et al., 1996) M. Billinghurst, S. Weghorst, T. Furness III. Shared Space: An Augmented Reality Approach for Computer Supported Collaborative Work, Extended abstract in Proc. of Collaborative Virtual Environments '(CVE'96), Nottingham, UK, 1996.
- (Billinghurst et al., 1998a) M. Billinghurst, J. Bowskill, M. Jessop, J. Morphet. A Wearable Spatial Conferencing Space, Proc. ISWC '98, pp. 76-83, 1998.
- (Billinghurst et al., 1998b) M. Billinghurst, S. Weghorst, T. Furness III. Shared Space: An Augmented Reality Approach for Computer Supported Collaborative Work, Virtual Reality: Virtual Reality - Systems, Development and Applications, 3(1), pp. 25-36, 1998.
- (Bray et al., 2000) T. Bray, J. Paoli, C. Sperberg-McQueen et al. Extensible Markup Language (XML) 1.0. <http://www.w3.org/TR/REC-xml/>, 2000.
- (Butz et al., 1998) A. Butz, C. Beshers, S. Feiner. Of Vampire Mirrors and Privacy Lamps: Privacy Management in Multi-User Augmented Environments. Proc. ACM UIST'98, pp. 171-172, Nov. 1998.
- (Butz et al., 1999) A. Butz, T. Höllerer, S. Feiner, B. MacIntyre, C. Beshers. Enveloping Computers and Users in a Collaborative 3D Augmented Reality, Proc. IWAR '99, pp. 1999.
- (Cruz-Neira et al., 1993) C. Cruz-Neira, D. Sandin, T. DeFanti. Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE. Proceedings of SIGGRAPH'93, pp. 135-142, 1993.
- (Eckel, 1998) G. Eckel. OpenGL Volumizer Programming Guide. SGI Inc., 1998.
- (Encarnação et al., 1999a) L. M. Encarnação, A. Stork, D. Schmalstieg, O. Bimber. The Virtual Table – A Future CAD Workspace. Proceedings of the 1999 CTS (Autofact) Conference, Detroit MI, Sept. 1999.
- (Encarnação et al., 1999b) L. M. Encarnação, O. Bimber, D. Schmalstieg, S. Chandler. A Translucent Sketchpad for the Virtual Table Exploring Motion-based Gesture Recognition. Computer Graphics Forum, pp. 277-286, Sept. 1999.
- (Encarnação et al., 2000) L. M. Encarnação, R. J. Barton III, P. Stephenson, P. Branco, J. Tesch, J. F. Mulhearn. Interactive exploration of the underwater sonar space in a semi-immersive environment. Submitted for publication, 2000.
- (Feiner et al., 1993) S. Feiner, B. MacIntyre, D. Seligmann. Knowledge-Based Augmented Reality. Communications of the ACM, vol. 36, no. 7, pp. 53-62, 1993.
- (Fuhrmann et al., 1998) A. Fuhrmann, H. Löffelmann, D. Schmalstieg, M. Gervautz. Collaborative Visualization in Augmented Reality. IEEE Computer Graphics & Applications, Vol. 18, No. 4, pp. 54-59, IEEE Computer Society, 1998.
- (Fuhrmann & Schmalstieg, 1999) A. Fuhrmann, D. Schmalstieg. Multi-Context Augmented Reality. Technical report TR-186-2-99-14, Vienna University of Technology, 1999. Available from <ftp://ftp.cg.tuwien.ac.at/pub/TR/99/TR-186-2-99-14Paper.pdf>
- (Goble et al., 1995) J. Goble, K. Hinckley, R. Pausch, J. Snell, N. Kassel. Two-Handed Spatial Interface Tools for Neurosurgical Planning. IEEE Computer, 28(7):20-26, 1995.

- (Goethe, 1808) J. W. von Goethe. Faust. Drama, first published 1808. English translation by D. Luke published by Oxford University Press, 1998.
- (Guiard, 1987) Y. Guiard. Assymmetric Division of Labor in Human Skilled Bimanual Action: The Kinematic Chain as Model. *Journal of Motor Behaviour*, 19(4):486-517, 1987.
- (Hesina et al., 1999) Hesina G., D. Schmalstieg, A. Fuhrmann, W. Purgathofer. Distributed Open Inventor: A Practical Approach to Distributed 3D Graphics, *Proc. VRST '99*, London, pp. 74-81, Dec. 1999.
- (Höllerer et al., 1999) Höllerer T., S. Feiner, T. Terauchi, G. Rashid, D. Hallaway. Exploring MARS: Developing indoor and outdoor user interfaces to a mobile augmented reality systems, *Computers & Graphics*, 23(6), pp. 779-785, 1999.
- (Ishii & Ulmer, 1997) Ishii H., B. Ulmer. Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms, *Proc. CHI '97*, pp. 234-241, 1997.
- (Kato et al., 2000) H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, K. Tachibana. Virtual Object Manipulation on a Table-Top AR Environment. *Proc. IEEE and ACM International Symposium on Augmented Reality*, 2000.
- (Kaufmann et al., 2000) H. Kaufmann, D. Schmalstieg, M. Wagner. Construct3D: A Virtual Reality Application for Mathematics and Geometry Education. To appear in: *Journal of Education and Information Technologies*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000.
- (Krüger et al., 1995) W. Krüger, C. Bohn, B. Fröhlich, H. Schüth, W. Strauss, G. Wesche. The Responsive Workbench: A Virtual Work Environment. *IEEE Computer*, vol. 28, no.7, pp. 42-48, 1995.
- (Mine et al., 1997) M. Mine, F. Brooks Jr., C. Sequin. Moving Objects in Space: Exploiting Proprioception In Virtual-Environment Interaction. *Proc. SIGGRAPH '97*, pp. 19-26, 1997.
- (Pausch et al., 1995) R. Pausch, T. Burnette, D. Brockway, M. Weiblen. Navigation and Locomotion in Virtual Worlds via Flight into Hand-Held Miniatures, *Proc. SIGGRAPH '95*, pp. 399-401, 1995.
- (Poupyrev et al., 1998) I. Poupyrev, N. Tomokazu, S. Weghorst. Virtual Notepad: Handwriting in Immersive VR. *Proc. of VRAIS'98*, 1998.
- (Raskar et al., 1998a) R. Raskar, G. Welch, M. Cutts, A. Lake, L. Stesin, H. Fuchs. The office of the future: A unified approach to image-based modeling and spatially immersive displays, *Proc. SIGGRAPH '98*, pp. 179-188, 1998.
- (Raskar et al., 1998b) R. Raskar, G. Welch, H. Fuchs. Spatially Augmented Reality. In: *Proceedings of the First IEEE Workshop on Augmented Reality (IWAR'98)*, San Francisco, CA, USA, A.K. Peters Ltd., 1998.
- (Reitmayr & Schmalstieg, 2000) G. Reitmayr, D. Schmalstieg. OpenTracker - An Open Software Architecture for Reconfigurable Tracking based on XML. To appear as a poster in: *IEEE Virtual Reality 2001*, Yokohama, Japan, March 13-17, 2001. Extended version available as technical report TR-186-2-00-18, Vienna University of Technology, June 2000.
- (Rekimoto & Saitoh, 1999) J. Rekimoto, M. Saitoh. Augmented Surfaces: A Spatially Continuous Workspace for Hybrid Computing Environments, *Proceedings of CHI'99*, pp.378-385, 1999.
- (Rekimoto, 1997) J. Rekimoto. Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments, *Proc. UIST '97*, pp. 31-39, 1997.
- (Schmalstieg & Schaufler, 1998) D. Schmalstieg, G. Schaufler. Sewing virtual worlds together with SEAMS: A mechanism to construct complex virtual environments. *Presence*, 8(4): 449-461, Aug. 1998.
- (Schmalstieg et al., 1996) D. Schmalstieg, A. Fuhrmann, Zs. Szalavari, M. Gervautz. Studierstube - Collaborative Augmented Reality, *Proc. Collaborative Virtual Environments '96*, Nottingham, UK, Sep. 1996.
- (Schmalstieg et al., 1999) Schmalstieg D., L. M. Encarnação, Zs. Szalavári. Using Transparent Props For Interaction With The Virtual Table, *Proc. SIGGRAPH Symp. on Interactive 3D Graphics '99*, pp. 147-154, Atlanta, GI, April 1999 (patent pending).
- (Schmalstieg et al., 2000) D. Schmalstieg, A. Fuhrmann, G. Hesina. Bridging Multiple User Interface Dimensions with Augmented Reality. *Proceedings of the 3rd International Symposium on*

- Augmented Reality (ISAR 2000), pp. 20-30, Munich, Germany, Oct. 5-6, 2000.
- (Shaw et al., 1993) C. Shaw, M. Green, J. Liang, Y. Sun. Decoupled Simulation in Virtual Reality with the MR Toolkit. *ACM Trans. on Information Systems*, 11(3):287-317, 1993.
- (Smith & Mariani, 1997) G. Smith, J. Mariani. Using Subjective Views to Enhance 3D Applications, *Proc. VRST '97*, pp. 139-146, New York, NY, Sep. 1997.
- (Stoev et al., 2000) S. Stoev, D. Schmalstieg, W. Strasser. Through-the-lens techniques for remote object manipulation, motion and navigation in virtual environments. Submitted for publication, 2000.
- (Stork & de Amicis, 2000) A. Stork, R. de Amicis. ARCADE/VT - a Virtual Table-centric modeling system. *Proc. of 4th International Immersive Projection Technology Workshop (IPT 2000)*, June 19-20, Ames, Iowa, 2000.
- (Strauss & Carey, 1992) P. Strauss, R. Carey. An object oriented 3D graphics toolkit, *Proc. SIGGRAPH '92*, pp. 341-347, 1992.
- (Szalavári & Gervautz, 1997) Zs. Szalavári, M. Gervautz. The Personal Interaction Panel - A Two-Handed Interface for Augmented Reality, *Computer Graphics Forum*, 16(3), pp. 335-346, Sep. 1997.
- (Szalavári et al., 1998a) Zs. Szalavári, A. Fuhrmann, D. Schmalstieg, M. Gervautz. *Studierstube - An Environment for Collaboration in Augmented Reality, Virtual Reality - Systems, Development and Applications*, 3(1), pp. 37-49, 1998.
- (Szalavári et al., 1998b) Zs. Szalavári, E. Eckstein, M. Gervautz. Collaborative Gaming in Augmented Reality Proceedings of VRST'98, pp.195-204, Taipei, Taiwan, November 2-5, 1998.
- (Tsao & Lumsden, 1997) J. Tsao, C. Lumsden. CRYSTAL: Building Multicontext Virtual Environments, *Presence*, 6(1), pp. 57-72, 1997.
- (Ulmer & Ishii, 1997) B. Ullmer, H. Ishii. The metaDESK: Models and Prototypes for Tangible User Interfaces. In *Proceedings of ACM UIST'97*, Banff, Alberta, Canada, pp. 223-232, 1997.
- (Ulmer & Ishii, 1998) B. Ullmer, H. Ishii, D. Glas. mediaBlocks: Physical Containers, Transports, and Controls for Online Media, *Proc. SIGGRAPH '98*, pp. 379-386, July 1998.
- (Underkoffler et al., 1999) J. Underkoffler, B. Ullmer, H. Ishii. Emancipated Pixels: Real-World Graphics in the Luminous Room. *Proc. SIGGRAPH 1999*, pp. 385-392, 1999.
- (Viega et al., 1996) J. Viega, M. Conway, G. Williams, R. Pausch. 3D Magic Lenses. In *Proceedings of ACM UIST'96*, pp. 51-58. ACM, 1996.
- (Weiser, 1991) M. Weiser. The Computer for the twenty-first century. *Scientific American*, pp. 94-104, 1991.
- (Wernecke, 1994) J. Wernecke. The Inventor Toolmaker : Extending Open Inventor, Release 2. Addison-Wesley, 1994.
- (Wloka & Greenfield, 1995) M. Wloka, E. Greenfield: The Virtual Tricoder: A Uniform Interface for Virtual Reality. *Proceedings of ACM UIST'95*, pp. 39-40, 1995.
- (Wohlfahrter et al., 2000) W. Wohlfahrter, L. M. Encarnação, D. Schmalstieg. Interactive Volume Exploration on the StudyDesk. *Proceedings of the 4th International Projection Technology Workshop*, Ames, Iowa, USA, June 19-20, 2000.

Windows on the World: 2D Windows for 3D Augmented Reality

Steven Feiner
Blair MacIntyre
Marcus Haupt
Eliot Solomon

Department of Computer Science
Columbia University
New York, NY 10027

212-939-7000

{feiner, bm, haupt, esolomon}@cs.columbia.edu

ABSTRACT

We describe the design and implementation of a prototype heads-up window system intended for use in a 3D environment. Our system includes a see-through head-mounted display that runs a full X server whose image is overlaid on the user's view of the physical world. The user's head is tracked so that the display indexes into a large X bitmap, effectively placing the user inside a display space that is mapped onto part of a surrounding virtual sphere. By tracking the user's body, and interpreting head motion relative to it, we create a portable information surround that envelopes the user as they move about.

We support three kinds of windows implemented on top of the X server: windows fixed to the head-mounted display, windows fixed to the information surround, and windows fixed to locations and objects in the 3D world. Objects can also be tracked, allowing windows to move with them. To demonstrate the utility of this model, we describe a small hypermedia system that allows links to be made between windows and windows to be attached to objects. Thus, our hypermedia system can forge links between any combination of physical objects and virtual windows.

KEYWORDS: augmented reality, virtual reality, virtual worlds, head-mounted displays, portable computers, mobile computing, window systems, X11, hypertext/hypermedia.

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

© 1993 ACM 0-89791-628-X/93/0011 ... \$1.50

INTRODUCTION

When we think of the use of head-mounted displays and 3D interaction devices to present virtual worlds, it is often in terms of environments populated solely by 3D objects. There are many situations, however, in which 2D text and graphics of the sort supported by current window systems can be useful components of these environments. This is especially true in the case of the many applications that run under an industry standard window system such as X [13]. While we might imagine porting or enhancing a significant X application to take advantage of the 3D capabilities of a virtual world, the effort and cost may not be worth the return, especially if the application is inherently 2D. Therefore, we have been exploring how we can incorporate an existing 2D window system within a 3D virtual world.

We are building an experimental system that supports a full X11 server on a see-through head-mounted display. Our display overlays a selected portion of the X bitmap on the user's view of the world, creating an X-based augmented reality. Depending on the situation and application, the user may wish to treat a window as a stand-alone entity or to take advantage of the potential relationships that can be made between it and the visible physical world. To make this possible, we have developed facilities that allow X windows to be situated in a variety of ways relative to the user and the 3D world.

In this paper we first present related work and provide an overview of our system. Next, we describe the different kinds of windows that we support, and show how these windows can be used to advantage by a simple hypermedia system. Finally, we explain the underlying system architecture and describe our current implementation.



Figure 1: User of our window system wearing a see-through head-mounted display. The large black cube and white triangle are the transmitters for two 3D tracking systems. Tracker receivers are worn on the head-mounted display, waist, and wrist.

RELATED WORK

Fisher et al. [7] describe a virtual environment in which the user can be presented with a collection of virtual information display windows and input control panels. Other groups have also built virtual world systems that support the creation of general purpose virtual control panels [2, 14]. In general, the extremely low resolution currently provided by the wide field-of-view, opaque, head-mounted displays used in most virtual environments projects has understandably discouraged researchers from porting or developing their own full-fledged window systems. Because the head-mounted display that we are using subtends a relatively small visual angle, what we sacrifice in field of view is compensated for by pixels that are sufficiently small to accommodate the detailed text and graphics displays that are typical of 2D window system applications.

We previously developed an X window manager that allows users to move regular X windows freely between a flat panel display and a much larger virtual surround in which it is embedded, presented on our see-through head-mounted display [5]. We refer to that system as a *hybrid user interface* because it merges two different kinds of display and interaction technologies. We run X on the flat panel and extended a regular window manager so that it maintains the virtual surround using a simple set of graphics routines that draw skeletal outlines of the windows with named title bars. The ideal approach would be to have a single X environment that seamlessly encompasses both

the flat panel display and virtual surround. However, the pixels on both displays are of different size and aspect ratio (our head-mounted display has nonsquare pixels), so the same window would need to be of different pixel resolution on each display for it to appear to be the same size. Since X supports only a single address space of uniformly sized pixels, we could not support a single X environment across the flat panel and the virtual surround without either customizing each client or interpreting all X commands at both resolutions.

The system described here differs in several significant ways from this previous work. First, we show how to provide full X functionality on a head-tracked, see-through head-mounted display. Building on top of X, we support three different kinds of windows, including ones fixed to the head-mounted display itself, ones fixed to a virtual surround that the user carries about, and ones fixed to movable objects in the 3D world. While our previous work tracked only head orientation relative to the flat panel display, our current system takes into account the position and orientation of the user's head and body, and of objects in the world. We have implemented our system using an efficient multilayer bitmap software architecture that composites bitmaps at interactive frame rates. Finally, we demonstrate our system with a hypermedia system that we have developed. (Other researchers have also suggested the benefits of incorporating hypermedia capabilities in virtual worlds [12, 8].)

While preparing the final version of this paper, we learned of two other ongoing projects that have goals similar to ours. Dykstra [4] has modified an X server so that its entire display can be texture-mapped on a 3D polygon displayed on a high-performance graphics workstation. Current hardware-supported texture-map size is quite small compared to typical X display resolution, however, and texture-map preprocessing is still too slow to support real-time modifications to the X display. Reichlen [11] uses a high-resolution, head-tracked, head-mounted display to index into a large X bitmap, much as we do, and achieves better real-time performance through the use of custom hardware. However, as in our earlier hybrid user-interface window manager, his system ignores head position. The user is assumed to be stationary, and is seated in a rotatable swivel chair in the center of the surround. Reichlen's head-mounted display is opaque, and his system makes no attempt to correlate windows with objects or positions in the surrounding 3D world.

SYSTEM OVERVIEW

As shown in Figure 1, our user wears a see-through head-mounted display, based on a Reflection Technology Private Eye 720×280 resolution display with a memory-mapped frame buffer. The display's bilevel red image is reflected by a mirror beamsplitter that merges the image with the user's view of the physical world. The head-mounted display is equipped with the receiver for a 3D tracking system that reports the position and orientation of the user's head, making it possible to change the information being presented based on this data. As described below, we also track the user's body and hand, and selected 3D objects. The white triangle and the large black cube in Figure 1 are the transmitters for two different 3D trackers: a Logitech Red Baron ultrasonic system and an Ascension Technology Extended Range Flock of Birds electromagnetic system. Using different tracker technologies allows us to trade off their relative advantages. For example, the ultrasonic system is not sensitive to the presence of metallic objects and magnetic fields, as is the magnetic system, whereas the magnetic system does not require a clear line of sight between transmitter and receiver, as does the ultrasonic system. (In Figure 1, the ultrasonic tracker is being used for the head, and the electromagnetic tracker for the body and hand.)

Because of the relatively small display, and our desire to present a large, encompassing environment, we take advantage of our head-tracking facilities and use the orientation of the user's head to index into a much larger information space than could be presented at one time on the display. This information space, the rectangular bitmap maintained by the X server, is mapped onto a portion of a sphere positioned about the user's head, just like the information surround of our earlier work [5]. To avoid confusing singularities, we use a relatively small portion of the sphere, roughly 170° longitude by 90° latitude, corresponding to a 6K by 2K bitmap (note that the display pixels are unfortunately nonsquare). As before, we make use only of yaw (rotation about an axis up through the neck, as in shaking the head "no," corresponding to the x coordinate, which is

mapped to longitude) and pitch (rotation about an axis through the ears, as in shaking the head "yes," corresponding to the y coordinate, which is mapped to latitude). We ignore roll (rotation about an axis from the front through the back of the head), which would require rotating the 2D bitmap to support. Thus, at any given time, the user sees an upright rectangular portion of the bitmap, providing a piecewise rectangular approximation to a spherical surround.

As described so far, in this model the absolute orientation of the user's head in the environment would determine which part of the surround is visible. Because our user is free to roam within the range of the tracking system, this model is often undesirable: the direction in which the user is facing would impose physical limits on how the head can be comfortably oriented, making it difficult to see parts of the surround without turning around. For example, since the window system's bitmap is mapped to a relatively small portion of the surround's sphere, if the user were facing in the "wrong" direction, information could be displayed only behind the user. To avoid this problem, we have also outfitted the user's body with an additional tracker positioned at their waist. We use the difference between the head-tracker and body-tracker orientation to determine which portion of the surround is mapped to the display. This models a surround that is fixed to the user's body, rather than to the world, a sort of virtual "portable desk" that is always in front of the user.

TYPES OF WINDOWS

We have developed support for three kinds of windows: surround-fixed, display-fixed, and world-fixed.

Surround-fixed windows. We refer to windows that are displayed at a fixed position within the surround as *surround-fixed* windows. These are the most commonly used windows in our system and are not intended to have a specific relationship to the physical world. As the user looks around, the portion of the surround (and its surround-fixed windows) that is visible changes.

Display-fixed windows. Quite a lot of head motion may be needed if we are interested in the relationships between two or more distant surround-fixed windows or if we would like to make frequent use of a particular surround-fixed window as we look around. Therefore, we have developed support for *display-fixed* windows that are positioned at a fixed location relative to the display itself, no matter how the user's head is oriented. (These windows would be the default if the entire bitmap—or the same part of it—were always mapped to the display.) A precedent exists for display-fixed windows within window managers such as *vtwm* [16] that support a virtual desktop that is larger than the physical display. In these systems, display-fixed windows, such as a control panel of window names, are implemented conventionally, whereas the illusion of a larger desktop is provided by actually moving the regular windows across the X bitmap as the user scrolls the display. Since we need to maintain a high frame rate, executing opaque moves for either kind of window is undesirable.

Therefore, we have implemented a compositing approach, described later, that overlays desired windows at specified locations in the coordinate system of the display.

World-fixed windows. Just as we may wish to fix some windows to the display, we may wish to fix others to locations or objects in the 3D world. We call these *world-fixed* windows. Our current implementation supports world-fixed windows by allowing users to specify a known object by name or a location by pointing in 3D. Taking into account the position and orientation of the user's head and the orientation of the user's body, a specified window is moved to an appropriate place in the X bitmap using regular X facilities (but see the conclusions). In general, if the user moves, the position in the bitmap must also change because it is a projection onto the virtual surround of a vector from the user's eye to the 3D window position. Furthermore, since we allow objects to be tracked, if the window is attached to a moving tracked object, the window must be moved as well. Since all windows, including world-fixed windows, are displayed by indexing into the X bitmap, each is always perpendicular to the user's direction of gaze and upright relative to the user's head.

A HYPERMEDIA APPLICATION

To demonstrate the utility of our system, we have developed a simple hypermedia application that supports the ability to make links between arbitrary X windows and to attach windows to objects and locations. To support the concept of linking as a universal system-wide resource [10], we use a display-fixed window for the hypermedia control panel, which allows us to make, break, and follow links. This assures that the control panel is always available wherever the user is looking.

Figure 2 shows a portion of the surround, directly in front of the user, who is looking at the two tracker transmitters. (This and all subsequent photographs were photographed directly through our see-through head-mounted display.) The wide display-fixed window at the bottom of the figure is the hypermedia control panel. The two other windows visible are a pair of *xeyes* and part of a weather map. Figure 3 shows another portion of the surround, seen from the same position, but a different orientation. Here we see the righthand side of the weather map and part of the manual entry for the program used to display it.

Linking two windows causes an arc to be drawn between them, and makes it possible for the link to be followed later to cause a linked window to be displayed and brought to the top of the window stack if it has been iconified or covered. The two windows in Figure 3 have been linked, as indicated by the diagonal arc drawn between them. We use a simple link manager that maintains a database of links between windows. To support linking to or from a physical object or 3D location, a transparent world-fixed window is associated with the object (as described later) and becomes the destination or source of the link, as appropriate. If the object is tracked, the window will move in the surround with the object, and the link will appear to follow the object. We have outfitted the user's hand with a 3D tracker.

To specify a 3D location, the user selects a button from a submenu of the control panel using the mouse and then points to a location by moving their tracked hand and clicking the mouse button.

An arbitrary application window can also be positioned at a desired location or relative to a (possibly tracked) object. In Figure 4, we have associated an *xpostit* [3] note with a tracked person, who is wearing a tracker around his neck, and an *xload* load-average meter with the corner of its computer's display. Figure 5 shows the same windows as seen from a different location and after the person has moved. Note that in both cases the windows remain perpendicular to the line of sight of the person wearing the head-mounted display.

IMPLEMENTATION

Our system architecture, shown in Figure 6, has six main components: the X server, the display server, the trackers, the world-fixed window server, the display-fixed window server, and the hypermedia application. Several additional utilities, not shown in the figure for clarity, are discussed later.

X Server

We modified a standard X11 R4 server, running under Mach [1], to use a virtual memory bitmap for the display, instead of the host machine's console. This allows us to create an arbitrarily large display, limited only by available memory. The X display bitmap is made available to our display server as a shared memory bitmap through the file system using the UNIX *mmap* facility. This is the only modification that we made to the server.

Display Server

The display server is written in C and runs under Mach on a 50 MHz Intel 486DX-based PC that supports the Private Eye display entirely in software. The display server was originally written to allow simple wire-frame and polygonal 3D graphics to be displayed on the Private Eye for use in the KARMA augmented reality system [6]. It has been enhanced to allow an arbitrary number of *overlays* to be placed on top of the original graphics display.

An overlay is defined by specifying a rectangular *viewport* on the Private Eye display, a *bitmap* to be overlaid, a 2D *offset* into the bitmap and a *raster operation* (e.g., *copy*, *xor*, *or*, etc.). The viewport-sized portion of the bitmap, whose upper left corner is specified by the offset, is overlaid onto the display by combining it with the image under the viewport using the specified raster operation. The redisplay process is optimized by creating a display list that takes advantage of the fact that certain raster operations are *opaque*, meaning they ignore the image under the overlay. As a result, only those portions of the original graphics screen and the overlay bitmaps that are actually visible in the final image need be examined. An overlay's index number indicates its priority relative to the other overlays.

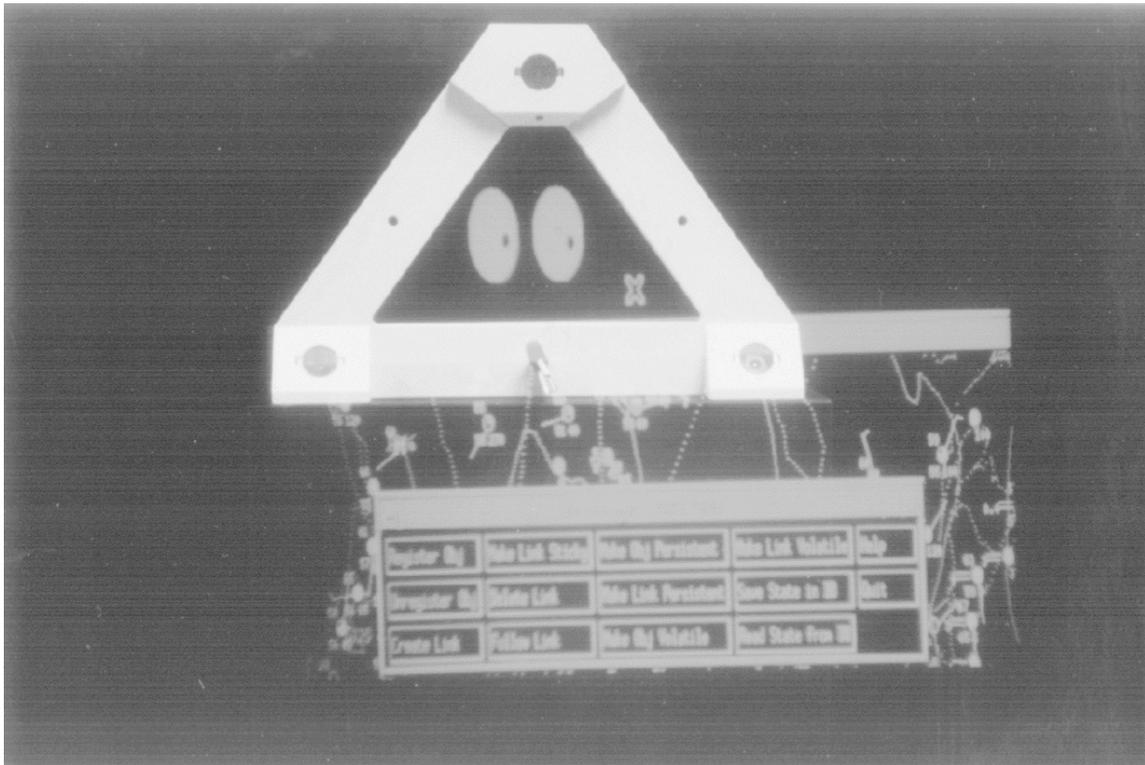


Figure 2: View of part of the surround seen through the head-mounted display.

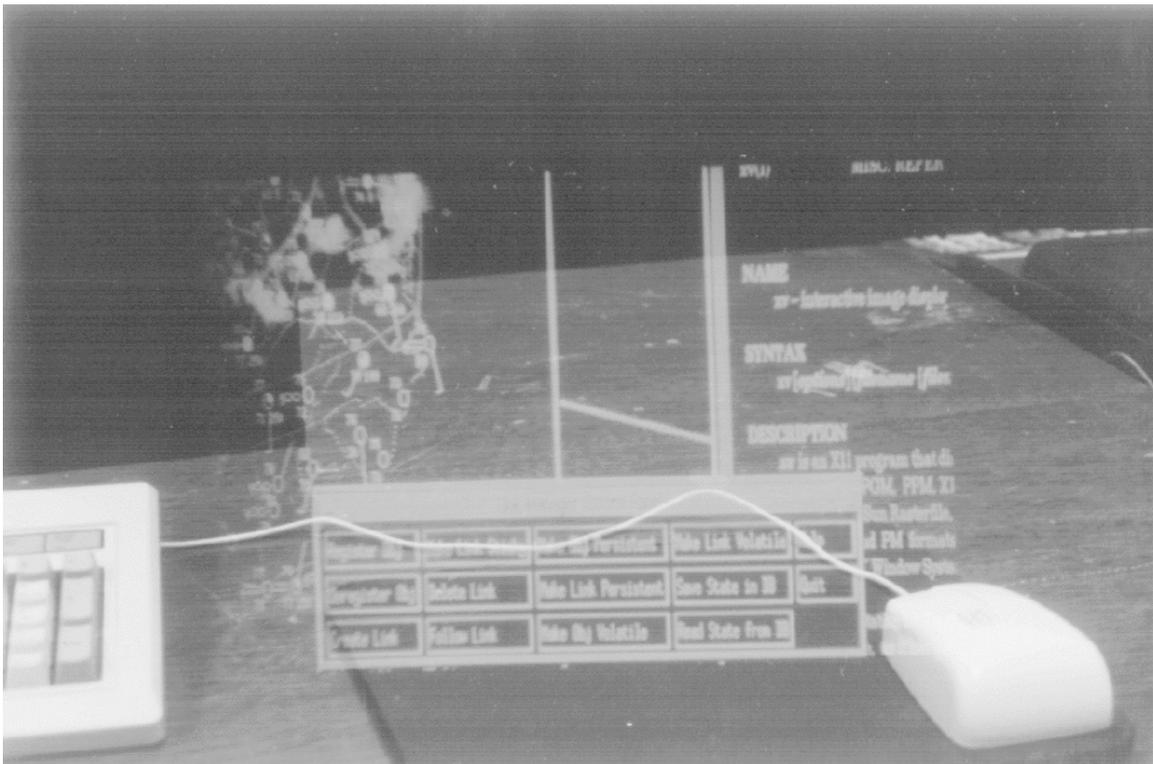


Figure 3: View of part of the surround from the same position as Figure 2, but a different orientation.

Further refresh optimizations are accomplished by noting that each write or read of the Private Eye's memory-mapped frame buffer involves a substantial number of wait

states. To avoid the overhead that would be associated with copying an entire new frame to the Private Eye's frame buffer each time, we maintain a buffer in main



Figure 4: A note is attached to a tracked person and a load-average meter to a 3D location.



Figure 5: The note and load-average meter of Figure 4 seen from a different position and after the person has moved.

memory that contains a copy of the Private Eye's frame buffer. Each word in the new frame is compared with the buffer, and those words that have changed from the pre-

vious frame are copied to the Private Eye's frame buffer.

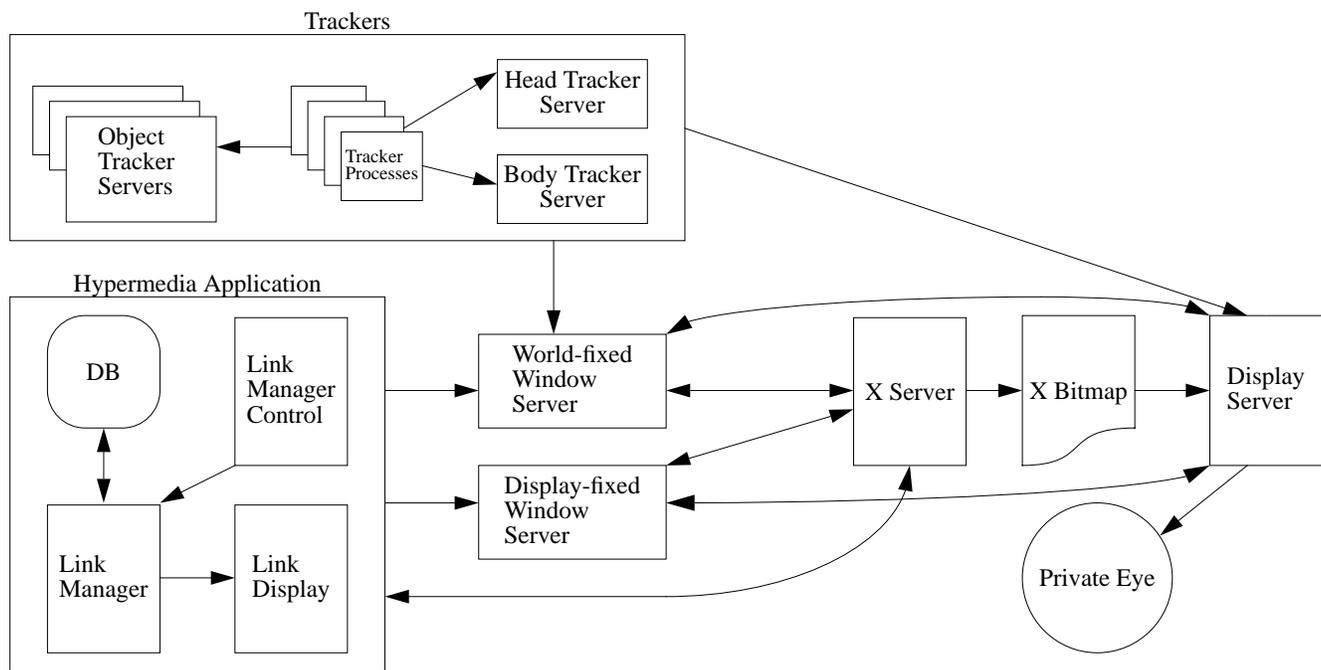


Figure 6: System architecture.

Our display list is organized so that all references to locations in the overlay bitmaps are relative to the 2D bitmap offset. This allows us to pan around an overlay bitmap without incurring the overhead of rebuilding the display list. For example, by having the head and body tracker servers store their position and orientation in the display server, the offset into the X bitmap overlay is recomputed each time the screen is refreshed, based on the user's head and body orientation, as discussed below.

Since the orientation values are changing constantly because of tracker noise and small movements of the user's head and body, the server performs *smoothing* and/or *thresholding* on the changes to the overlay offset resulting from head and body motion. In both cases, the difference between the current offset and the newly computed offset is examined. With *smoothing*, if this difference is less than a specified amount, the offset change is spread over successive refresh frames. *Thresholding*, on the other hand, simply ignores the new offset if the difference is below a specified threshold.

Smoothing, as the name implies, was designed to smooth out small changes and thus eliminate the jitter caused by tracker noise and small head and body movements. However, because it constantly adjusts the offset in the bitmap by a small amount, smoothing adversely affects the performance of the double buffering scheme. Thresholding attempts to eliminate jitter by ignoring small changes. However, if the threshold is sufficiently large to remove the jitter caused by tracker noise, intentional small head movements are ignored. A combination of the two techniques seems to work best. When combined with Kalman filtering of the head tracker motion (see the section on trackers), the image being viewed remains extremely stable when the

user is not moving, but is responsive to large movements.

Projecting the Surround

There are two similar projections of interest here. First, we need to project the appropriate part of the X bitmap onto the head-mounted display to create the illusion of a virtual surround. Second, we need to project world-fixed windows onto the X bitmap so that they appear in the correct place on the virtual surround.

As mentioned above, the head-tracker and body-tracker servers store their position and orientation in the display server so that the offset into the X bitmap overlay may be recomputed each time the screen is refreshed. Computing the offset from these values is a two-step process. First, the user's *view direction vector* is determined by applying two transformations to the z axis: the *surround viewing transformation* and the *centering transformation*. The surround viewing transformation is the composition of the head-tracker orientation and the inverse body-tracker orientation and gives us the direction of the user's head relative to their body. The centering transformation is initially the identity transformation. Whenever the user requests that the virtual surround be centered about the current viewing direction, the system saves the inverse of the current surround viewing transformation as the new centering transformation. Second, the view direction vector is converted to polar coordinates. The two angular components are multiplied by the number of pixels per degree of horizontal or vertical visual angle (determined for each user during calibration) to compute the offset into the X bitmap (the *surround view offset*).

Projecting world-fixed windows onto the X bitmap is accomplished similarly, with two differences. First, the *window direction vector* (analogous to the view direction vector) is determined by applying two transformations to the vector from the viewer's eye to the window's 3D position: the *window transformation* (analogous to the surround viewing transformation) and the *centering transformation*. The window transformation is simply the inverse body-tracker orientation, as the orientation of the user's head does not affect the projection of an object onto the surround. The centering transformation is the same as before. The second difference is that we can no longer ignore the third orientation component of the user's head, the *roll* or twist along the z axis. To account for this, we rotate the 2D projection of a window on the surround around the current surround view offset by the inverse of the roll of the user's head.

Trackers

The body and object servers, and the associated lower-level tracker processes, are written in C. We typically run them on other workstations to avoid imposing a large load on the machine that runs the X server and display server. The tracker servers provide a uniform, high-level interface to the different tracking systems. All trackers report their position and orientation to the world-fixed window server. The head and body servers also update their position and orientation in the display server.

We use a Kalman filter [9] in the head tracker to smooth the motion and decrease lag. The head server is also calibrated to report the position of the user's eye, as opposed to the position of the physical tracker.

Display-Fixed Window Server

Display-fixed window support consists of two separate components: the compositing of a specified area of the X bitmap into a fixed area viewport of the display and control of the X cursor to provide the user with the illusion that the display-fixed windows are actually where they appear to be on the X display.

Compositing the windows onto the display is handled by placing the X windows in a portion of the X bitmap not visible anywhere on the virtual surround and adding two display-sized overlays in the display server with priorities higher than that of the X bitmap's overlay. The first of these overlays is used to mask out those portions of the display containing display-fixed windows. Its bits are zero at pixels occupied by a display-fixed window and one at all other pixels. The second overlay is used to *or* display-fixed windows on to the display. It references the portion of the X bitmap in which the display-fixed windows reside. Together, both overlays create the appearance of opaque display-fixed windows. The mask overlay is contained in shared memory and is updated by the display-fixed window server to reflect the current window structure of the display-fixed portion of the X bitmap.

To handle the interaction of the cursor with the display-fixed windows, the display-fixed window server watches all X cursor motion. When the cursor enters a portion of the display in which a display-fixed window is visible, the cursor is warped to the actual window position in the X bitmap. When the cursor leaves the actual window boundaries, the cursor is warped back to the appropriate visible part of the user's display. This is accomplished quickly by testing the appropriate bit in the mask overlay. To the user, it simply appears that the cursor moves in and out of the display-fixed windows.

This solution is sufficient for most applications. However, it falls short of providing the user with a completely transparent implementation of display-fixed windows. For example, display-fixed windows do not interact with the window manager properly since their visibility priority is always higher than that of any surround-fixed or world-fixed windows. While it would be simple to implement a scheme where the stacking order was taken into consideration when creating the mask overlay, allowing display-fixed windows to appear behind world-fixed and surround-fixed windows, we chose not to do so for performance reasons. Another shortcoming is the result of our decision to suspend cursor warping when any mouse button is depressed. This was done to prevent unexpected results from occurring during window manager operations such as window moving and resizing. While the results are reasonable in most situations, there are occasional surprises, such as when the cursor disappears behind a display-fixed window while moving a surround-fixed window.

World-Fixed Window Server

The main responsibility of the world-fixed window server is to maintain a database of known objects in the physical world. The object state information retained by the server includes whether or not the object is tracked, its current 3D location, and the 2D X address of a small, transparent, input-only *proxy window* created by the world-fixed window server. Proxy windows provide client applications with a convenient method to determine quickly the projection of a physical object on the X display. The server continuously updates the position of an object's proxy window based on the user's head and body positions and the position of the object. Additionally, the server allows clients, such as our hypermedia application, to attach X windows to each object. The server ensures the location of each attached X window is consistent with the projection of its associated object on the virtual surround.

Hypermedia Application

The hypermedia application consists of a link manager, link database, link manager control, and link display facility.

The *link manager control* provides an interface to the linking subsystem via the *link manager control panel*, which is presented as a display-fixed window. Using the control panel, arbitrary, hypertextual links may be placed between any two windows or physical objects in the virtual surround. Basic features include following links and deleting

links, and persistent storage of objects and links in the *link database*.

The engine for the linking subsystem is the *link manager*. The link manager processes the actions requested by the user via the link manager control panel. The link manager internally maintains state information of all objects and links in the user's environment. This information may be read from or stored into the link database to provide consistency between sessions.

Links between windows and/or objects (represented by their proxy windows) are indicated by drawing an arc between them. This function is carried out by the link manager. The arc is refreshed whenever either of its endpoints is moved and is removed when the window associated with either of its endpoints exits. Currently, links are drawn directly upon the root window. The link line is not drawn over inferiors of the root window. No damage events occur in any window other than the root window when link lines are added or removed.

Miscellaneous Utilities

Unlike our previous work [5], no changes to the window manager are required.¹ We currently run an unmodified version of the *mwm* window manager. To accomplish additional window-manager-like activities, such as moving the cursor to the center of the area currently being displayed by the display server, we bind simple utility programs to function keys using the window manager. These utilities contact the various servers, depending on their purpose.

For example, to move the cursor to the center of the displayed area, the *movetocenter* program queries the display server to determine the part of the X bitmap that is visible, and moves the cursor to the center of this region. The *setthreshold* and *setsmoothing* utilities adjust the smoothing and threshold values in the display server. The *centerview* program contacts the display server to center the virtual surround about the user's current viewing direction. The *togglefixed* program contacts the display-fixed window server to toggle a selected window in and out of display-fixed mode.

Performance

We currently achieve between 6 and 20 frames per second, double-buffered, for the figures in this paper, which use three overlays (one for the main part of the surround, and two for the display-fixed windows, as described in the section on the display-fixed window server). The exact frame rate depends on whether or not there are display-fixed or world-fixed windows. If there are no display fixed windows, the display-fixed window overlays are not needed, increasing the frame rate about 10 frames per second. If

there are world-fixed windows, they must be continuously moved, forcing the X server (which runs on the same processor as the display server) to consume a significant amount of processing time. Unfortunately, with even one world-fixed window, the amount of moving and redrawing performed by the X server is substantial, reducing the frame rate from 18–20 to as low as 6–10 frames per second.²

CONCLUSIONS AND FUTURE WORK

We have described an approach to presenting full X window system functionality on a head-tracked, head-mounted display. Minimal server modifications were needed to allow the server to create an arbitrarily-sized X bitmap and make it accessible to others. We developed a fast software display server that supports multiple overlaid bitmaps and the ability to index into and display a selected portion of a larger bitmap. Coupled with tracking of the user's head, body, and hand, and objects in the world, we used this to support windows that were fixed to the display, to an information surround, and to the 3D world.

An important question to ask about any user interface that uses both experimental hardware and software is what stands between the current implementation and a practical system? We see a number of practical limitations, that we expect will be overcome during the next five to ten years. Our head-mounted display, although relatively lightweight (14 oz.) compared to commercial opaque systems is still relatively heavy and socially unacceptable in appearance. Its image is dim and small (22° horizontal field of view). Although its focus is user-settable, it can only be adjusted manually, limiting what can be in focus in the physical and virtual worlds simultaneously. The short range and relative inaccuracy of the 3D trackers restricts the workspace within which a user can roam (currently a 12' square), as does the tether from our display and trackers to what are currently nonportable workstations. We note, however, that our current frame rate is quite comfortable, yet is supported entirely in software on an inexpensive commodity personal computer.

There are a number of directions in which we are interested in taking this work. For example, in theory, display-fixed and world-fixed windows could both be implemented by the same mechanism (both through X or through overlays). However, in our current system, as mentioned above, initial set-up time is required each time an overlay is added, deleted, or changed in location or size. Therefore, changing the location of an overlay dynamically has a significant transient impact on the frame rate. We expect to improve our implementation to reduce this impact, however, and are interested in comparing the two approaches and their impact both on performance on the user interface. Compositing is clearly fastest, but does not behave like X in the sense that each overlay maintains its priority. Using X to do the moving means that the windows act like X windows,

¹Nevertheless, there are several places in which window-manager modifications would be extremely helpful (e.g., to assure that window-manager dialogue boxes always appear in a visible location) [11].

²We expect to improve this significantly by using thresholding when tracking world-fixed windows.

but damage repair can exact a significant performance penalty. One particularly interesting prospect would be to write an X window manager used our display manager to support all window movement operations through real-time compositing. (A commercially available example of this approach was the Lexidata Lex90's hardware window system, developed in the early 1980's.)

Our current support for showing links between windows is extremely unsatisfactory since they are drawn only on the root window. Although we could refresh them whenever they are overwritten by window movement, we instead intend to support link display by means of an additional overlay that is *or*'ed on top of our other overlays. Rather than using the stand-alone 3D graphics primitives supported by the display server, this overlay could be the bitmap of a separate output-only X server.

One of our most important directions will involve incorporating our X support with the knowledge-based 3D graphics generated by our KARMA augmented reality system [6]. This will make it possible to integrate 2D windows with virtual as well as physical objects. The display software facilities to support this are already in place since the display server is an enhanced version of the server originally developed for KARMA. We are also adding 3D spatial sound [15] to KARMA, and expect to explore its implications for our X environment. For example, X activity in windows that are not currently within the user's field of view may be indicated by appropriately positioned sonic cues intended to direct the user's attention in the appropriate direction.

ACKNOWLEDGMENTS

Research on this project is supported in part by the Office of Naval Research under Contract N00014-91-J-1872, a gift and summer internship from NYNEX Science & Technology, the New York State Center for Advanced Technology in Computers and Information Systems under Contract NYSSTF-CAT-92-053, the Columbia Center for Telecommunications Research under NSF Grant ECD-88-11111, NSF Grant CDA-92-23009, and equipment grants from the Hewlett-Packard Company and Ascension Technology. Thomas Magdahl built the camera mount used to take pictures through our head-mounted display.

REFERENCES

[1] M. Accetta, et al.
Mach: A New Kernel Foundation for UNIX Development.
In *Proc. Summer Usenix*, pages 93–112. July, 1986.

- [2] Butterworth, J., Davidson, A., Hensch, S., and Olano, T.
3DM: A Three Dimensional Modeler Using a Head-Mounted Display.
In *Proc. 1992 Symp. on Interactive 3D Graphics (Special Issue of Computer Graphics)*, pages 135–138. Cambridge, MA, March 29–April 1, 1992.
- [3] Curry, D.
Xpostit: X window System Post-It Notes (UNIX Man Page).
West Lafayette, IN, 1991.
- [4] Dykstra, P.
X11 in Virtual Environments.
In *Proc. IEEE 1993 Symposium on Research Frontiers in Virtual Reality*. San Jose, CA, October 25–26, 1993.
- [5] Feiner, S. and Shamash, A.
Hybrid User Interfaces: Breeding Virtually Bigger Interfaces for Physically Smaller Computers.
In *Proc. UIST '91 (ACM Symp. on User Interface Software and Technology)*, pages 9–17. Hilton Head, SC, November 11–13, 1991.
- [6] Feiner, S., MacIntyre, B., and Seligmann, D.
Knowledge-Based Augmented Reality.
Communic. ACM 36(7):52–62, July, 1993.
- [7] Fisher, S., McGreevy, M., Humphries, J., and Robinett, W.
Virtual Environment Display System.
In *Proc. 1986 Workshop on Interactive 3D Graphics*, pages 77–87. Chapel Hill, NC, October 23–24, 1986.
- [8] Fitzmaurice, G.
Situated Information Spaces: Spatially Aware Palmtop Computers.
Communic. ACM 36(7):38–49, July, 1993.
- [9] Liang, J., Shaw, C., and Green, M.
On Temporal-Spatial Realism in the Virtual Reality Environment.
In *Proc. UIST '91 (ACM Symp. on User Interface Software and Technology)*, pages 19–25. Hilton Head, SC, November 11–13, 1991.
- [10] Meyrowitz, N.
The Missing Link: Why We're All Doing Hypertext Wrong.
In Barrett, E (editor), *The Society of Text: Hypertext, Hypermedia, and the Social Construction of Information*, pages 107–114. MIT Press, Cambridge, MA, 1989.
- [11] Reichlen, B.
Sparcchair: A One Hundred Million Pixel Display.
In *Proc. IEEE VRAIS '93*. Seattle, WA, September 18–22, 1993.

- [12] Robinett, W.
Synthetic Experience: A Taxonomy.
Presence 1(2):229–247, Spring, 1992.
- [13] Scheifler, R., and Gettys, J.
The X Window System.
ACM Trans. on Graphics 5(2):79–109, April, 1986.
- [14] Shaw, C., Green, M., Liang, J., and Sun, Y.
Decoupled Simulation in Virtual Reality with the
MR Toolkit.
ACM Trans. on Information Systems 11(2), July,
1993.
- [15] Wenzel, E., and Foster, S.
Realtime Digital Synthesis of Virtual Acoustic En-
vironments.
In *Proc. 1990 Symp. on Interactive 3D Graphics*
(*Computer Graphics*, 24:2, March 1990), pages
139–140. Snowbird, UT, March 25–28, 1990.
- [16] Williams, N. and Edmondson, D. (with LaStrange,
T.).
vtwm documentation.
1990.

Exploring MARS: Developing Indoor and Outdoor User Interfaces to a Mobile Augmented Reality System

Tobias Höllerer, Steven Feiner, Tachio Terauchi, Gus Rashid,
Drexel Hallaway

Dept. of Computer Science, Columbia University, New York, NY 10027

Abstract

We describe an experimental mobile augmented reality system (MARS) testbed that employs different user interfaces to allow outdoor and indoor users to access and manage information that is spatially registered with the real world. Outdoor users can experience spatialized multimedia presentations that are presented on a head-tracked, see-through, head-worn display used in conjunction with a hand-held pen-based computer. Indoor users can get an overview of the outdoor scene and communicate with outdoor users through a desktop user interface or a head- and hand-tracked immersive augmented reality user interface.

Key words: Augmented Reality. Wearable Computing. Mobile Computing. Hypermedia. GPS.

1 Introduction

As computers increase in power and decrease in size, new mobile and wearable computing applications are rapidly becoming feasible, promising users access to online resources always and everywhere. This new flexibility makes possible a new kind of application—one that exploits the user's surrounding context.

Location-aware computing [2] allows us to link electronic data to actual physical locations, thereby augmenting the real world with a layer of virtual information. Augmented reality [1] is particularly well suited as a user interface (UI) for location-aware applications. Equipped with location and orientation sensors and with a model of the user's environment, the computer can annotate the user's view of the physical world [6]. Through optical or video-mixed see-through displays, the user views the electronic information in situ, attached to the physical world,



Fig. 1. A view through the mobile user's see-through head-worn display. (This and all other augmented reality images in this paper were captured by a video camera aimed through the see-through head-worn display.)

and can interact with this virtual layer or even modify it. Thus, the world *becomes* the UI.

A wearable UI alone will not be enough to fully capture the potential of a world-wide layer of spatialized information. For various tasks, a stationary computer system will be more adequate, especially for those applications whose UIs work best with physically large displays. Among these applications are tools, especially collaborative ones, for authoring the information layer, for obtaining a broad-scale overview of relevant information, and for playing back logs of user interactions with the augmented world.

In this paper we present a mobile augmented reality system (MARS) testbed that employs four different UIs:

- (1) Outdoors, a mobile user, tracked by a centimeter-level real-time-kinematic global positioning system (GPS) and an inertial/magnetometer orientation sensor, and equipped with our prototype backpack computer system, experiences the world augmented by multimedia material displayed on a see-through and hear-through head-worn display (cf. Figures 1 and 2).
- (2) A hand-held display offers a map-based UI to some of the virtual information (Figure 3), either in conjunction with the backpack or standalone.
- (3) Indoors, a desktop or projection-display UI (Figure 4a), based on a 3D environment model, lets users create virtual objects and highlight and annotate real objects for outdoor users to see, and maintain histories of outdoor users' activities; in turn, outdoor users point out interesting objects and events for indoor users to view.
- (4) An immersive version of the indoor UI (Figure 4b) relies on see-through head-worn displays, in conjunction with 6DOF head and hand trackers, and 3DOF object trackers, to overlay and manipulate virtual information on and over a physical desk.



Fig. 2. A user wearing our prototype MARS backpack.

1.1 Related Work

Many researchers have addressed the development of outdoor location-aware mobile and wearable systems. Some have relied on modifying the environment being explored; for example, Smailagic and Martin [15] label campus information signs with bar codes to provide location-specific information on a hand-held computer equipped with a bar code scanner. Others have combined GPS and orientation trackers to produce map-based contextual displays [10], to provide audio navigation assistance to blind users [11], or to annotate the world with overlaid textual labels [5,19,9] or multimedia information [8]. These projects presage the goal articulated in Spohrer's proposal for a "WorldBoard" [16]: a world-wide spatial hypertext of information anchored to physical locations and objects.

Indoors, researchers have begun to explore the development of multi-user augmented reality systems. Szalavari and colleagues [18] use tethered trackers to support a collaborative augmented reality environment. Billinghamurst and colleagues [3] rely on visual fiducials to position texture-mapped representations of participants in an augmented reality teleconference. Rekimoto and Saitoh [14] use a projection display to show links among material that is displayed on the optically tracked computers of a meeting's participants, while Szalavari, Eckstein, and Gervautz [17] and Butz and colleagues [4] present shared context on see-through head-worn displays whose customized overlays support privacy.

2 System Overview

The mobile part of our MARS environment is explored by a roaming user wearing our backpack system. The user is restricted to the area around Columbia's campus that satisfies three conditions:



Fig. 3. The hand-held computer with the map UI.

- Within range of the local base station for our real-time-kinematic GPS system, which sends error correction information that makes possible centimeter-level position tracking of roaming users.
- Covered by our wireless communications infrastructure: a network of spread-spectrum radio transceivers that give us access to the internet.
- Represented within our 3D environment model: a coarse block representation of all Columbia buildings, pathways, and main green spaces. Our model also includes selected underground infrastructure and several buildings that have been demolished (cf. Figure 7).

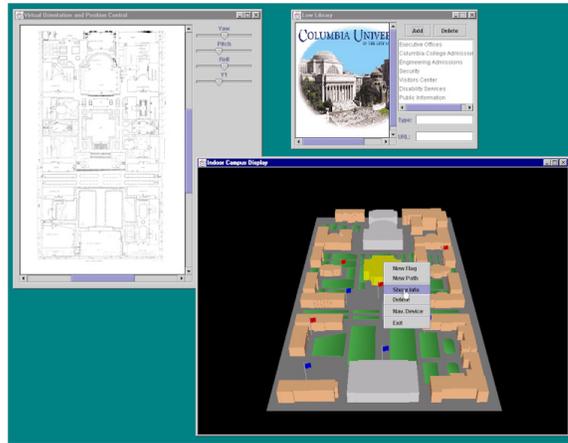
In contrast, our indoor UIs are constrained only by the third condition: the 3D environment model.

2.1 Functionality

One goal of our research is to explore the kinds of UIs that will be needed to interact with a spatialized hypertext, whose multimedia information can be linked to physical objects and tracked users. We support text, audio, static images, video, 3D graphics, 360° surround view images, and Java applets (cf. [8]). We are especially interested in *hybrid UIs* [7] that combine different display technologies, such as the tablet computer and see-through head-worn display worn by the outdoor user of Figure 2. Thus far, we have developed two prototype applications for outdoor users: a campus tour [5] and a journalistic storytelling system [8].

For indoor users, we have explored three areas of core functionality:

- (1) Creating and editing virtual information, and associating it with real objects and locations.
- (2) Obtaining an overview and keeping a log of outdoor users' activities. This could be done by the user herself, to review the history of her whereabouts and



a)



b)

Fig. 4. Indoor UIs. a) Desktop UI, showing (clockwise from lower right) main model window, map window, information window. b) Immersive augmented reality UI. The user positions a virtual flag by moving its position-tracked physical base.

interactions with the world, or by somebody who is supervising the outdoor user.

- (3) Communicating among indoor and outdoor users, including giving status reports and providing guidance.

2.2 System Architecture

Figure 5 shows an overview of the system architecture for our MARS environment. A detailed description of our backpack system's hardware design can be found in [5] and [8]. We use Sony LDI-100B and LDI-D100B 800×600 triad resolution, color, see-through, head-worn displays for indoor and outdoor augmented reality UIs. Our current hand-held computer, shown in Figure 3, is a Mitsubishi AMiTY CP, with a Pentium MMX 166 MHz CPU, running Windows95. Our wireless communication infrastructure comprises Lucent WavePoint II base stations and Wave-

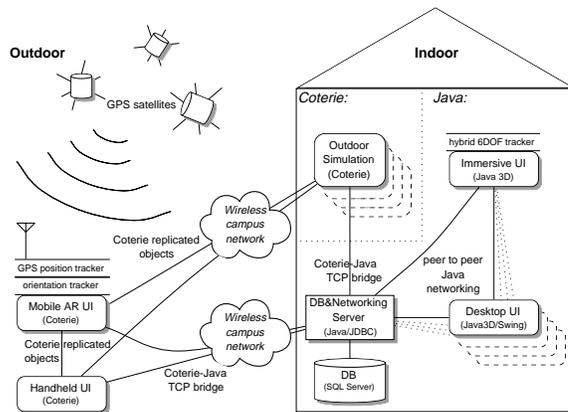


Fig. 5. MARS architecture.

LAN PC cards for our mobile computers. Our indoor immersive UI is tracked with an InterSense IS600 Mark II hybrid ultrasonic and inertial 6DOF tracker.

We use two different software development platforms for our outdoor and indoor applications: our Coterie distributed virtual environment infrastructure [12,13], based on Modula-3, for the outdoor backpack system and its indoor simulation version, and Java/Java 3D for the indoor desktop and immersive UIs, as well as for the main database interface. Our hand-held-based map UI is coded in Coterie. Alternatively, we can run a custom HTTP server (coded in Coterie) on the hand-held computer and use it with a generic web browser.

All applications in our testbed access a main database that contains a model of the physical environment and of the virtual information added to it. When each UI is started up, it reads the most recent state of the database. Internally, the data is organized in a relational format, currently maintained using Microsoft SQL Server.

A database server process, written in Java using the JDBC API, provides client processes (multiple users and UIs) with access to this data. To make this possible, we first developed a client-server database access protocol. Not surprisingly, the latency of these calls is too great for real-time graphics updates (e.g., rendering a moving outdoor user in an indoor system). To address this, we developed a simple UDP-based (in the Coterie–Java link: TCP-based) peer-to-peer communication infrastructure, emulating a distributed shared memory model for the objects that need to be updated rapidly. We are currently exploring several options to replace this with a more general distribution scheme based on object replication, similar to what is used in our Coterie system [12].

2.3 The Development Environment

We developed several tools and techniques to make the development and testing of new collaborative outdoor and indoor UIs easier and more efficient.



Fig. 6. An in-place menu realized with a screen stabilized menu and a leader line.

To test new outdoor UI components without actually taking the backpack system outside, we designed a Coterie application that simulates an outdoor user indoors. This program can be run in two modes: free navigation mode, which supports mouse-based navigation over the whole terrain on a conventional CRT display, using controls similar to those of first-person action games, and immersive mode, which uses the same head-worn display and orientation tracker we use outdoors. In immersive mode, we assume a fixed position in our environment and use either a 360° omnidirectional image taken from that position as a “backdrop” [8] or display the 3D environment model.

Since both real and simulated outdoor users are treated alike in their interaction with other processes, we can do tests with multiple roaming users in this fashion, although we currently have only one physical backpack system.

We developed a separate authoring tool for creating and placing new 3D models. It uses a 2D map of the area to be modeled, with its latitude–longitude coordinates. The user can trace over the 2D map with the geometrical primitives typically supported by 2D drawing programs. The tool can extrude these outlines in 3D to create simple models of buildings that are saved for use in our MARS environment.

3 UI Design

3.1 Outdoor MARS UI

Building on our first MARS work on overlaying labels on campus buildings [5], our head-worn UI, shown in Figures 1, 6, 7, and 8 (b–c), consists of a world-stabilized part and a screen-stabilized part. World-stabilized items, which are visually registered with specific locations, and displayed in the correct perspective for the user’s



Fig. 7. 3D model of a building that once occupied Columbia's campus, overlaid on its former site.

viewpoint, include labels of buildings (Figure 1), iconic flags representing important information that can be further examined by the user (Figure 1), and virtual representations of physical buildings (Figure 7).

Screen-stabilized items are fixed to the display and are always visible; they include the menu bar at the top, and the cone-shaped pointer at the bottom. Head-worn display menus are controlled through a trackpad mounted on the hand-held display. The cone-shaped pointer indicates the currently selected object. An object can be selected through several mechanisms, including an approximation to gaze-directed selection, and following links presented in the on-screen menu or on the pen-based hand-held computer. When enabled, our approximation to gaze-directed selection is accomplished by the user orienting her head so the desired object's projection is closer than any other to the center of the head-worn display and within a small target area. If it remains the closest within that area for a half second, the object's label will smoothly change color over that interval to confirm the selection.

Some items are partially world-stabilized and partially screen-stabilized. For example, the vertical menu at the left of Figure 6 displays a set of multimedia items associated with the flag at the center of the display. The menu is screen-stabilized and the flag is world-stabilized; however, the leader line connecting them has a screen-stabilized vertex attached to the menu and a world-stabilized vertex attached to the flag. The leader line helps establish the relationship between the menu and flag, even when the flag is not visible because the user is turned away from it. (A detailed discussion of the UIs for our two outdoor MARS applications can be found in [5] and [8].)

3.2 *Hand-held Map UI*

The map UI, running on our hand-held computer, can be used in conjunction with our outdoor MARS UI or standalone. The “map” is a projection of our 3D environment model, which can be chosen to emulate a conventional 2D map (Figure 3). The user can review her current position and select objects; when used in conjunction with the outdoor MARS UI, an object selected on the map will be selected on the head-worn display (and vice versa), so the user can be directed toward it.

3.3 *Indoor UIs*

We have two indoor UIs: a desktop UI and an immersive augmented reality UI.

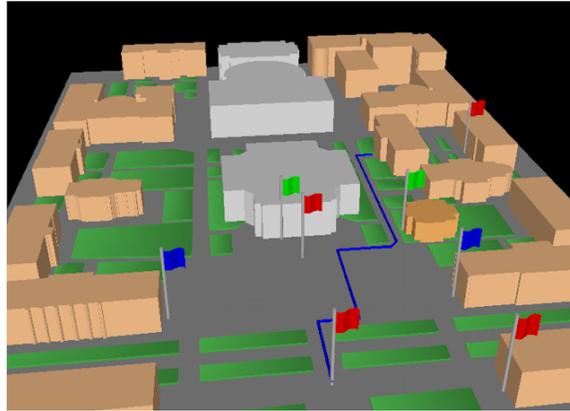
The desktop UI presents information in multiple windows, as shown in Figure 4 (a). The main window shows a navigable 3D model of the campus environment in which the user can select objects to extract information and create new objects. A 2D map window, shown at the left, can be used to aid in navigating the 3D view.

The main window’s pop-up menu allows users to create new objects, such as flags and paths through the environment, to delete objects, and to bring up an information window for any given object. An information window, shown at the top right of Figure 4 (a) makes it possible to view and modify all information associated with its object for display to MARS users.

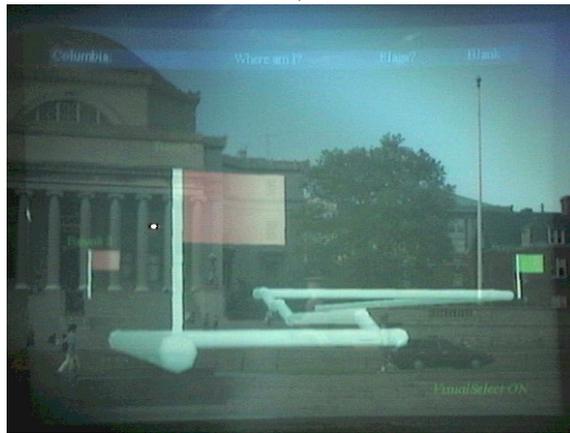
Users of the immersive augmented reality UI wear see-through head-worn displays tracked by an InterSense IS600 Mark II 6DOF tracker. The 3D environment’s ground plane is coplanar with a physical desk. Our main input devices are Logitech wireless trackballs, tracked by wireless InterSense position sensors. We also use the position sensors as physical props. For example, Figure 4 (b) shows a user placing a virtual flag by moving a position sensor that acts as the physical base for the flag.

3.4 *Indoor/Outdoor Interaction*

Our four UIs offer many opportunities for indoor/outdoor communication and collaboration. New virtual objects can be introduced by any UI and, when moved around, their position is updated in all participating UIs. This can be used, for example, to highlight points of interest. Figure 8 shows an example of indoor/outdoor interaction: an indoor user giving guidance to a roaming user by drawing a path on the virtual model (part a). Parts (b) and (c) shows outdoor views of that path as seen from two different perspectives.



a)



b)



c)

Fig. 8. Outdoor paths. a) Creating a path in the desktop UI. b) Same path, seen outdoors from ground level. c) Same path, seen from above.

4 Conclusions and Future Work

We have described our experimental MARS testbed, presenting four different UIs for indoor and outdoor users, enabling users to annotate the real world with vir-

tual information and to explore the merged environment. Indoor users can provide guidance to outdoor users by sketching paths or pointing out objects of interest. An outdoor user's position and head orientation can be tracked in the indoor system and logged in the main database for later review.

There are many directions in which we would like to extend this work. Because the indoor augmented reality UI lends itself well to collaborative work, we are integrating it with our EMMIE multi-user augmented reality system [4]. This will allow us to further explore EMMIE's support for customized views for displaying private information; for example, to allow different indoor users to direct and track their own separate crews of outdoor users. Over the past few months, we have started to collaborate with researchers at the Naval Research Lab, with the goal of linking our systems with ones that they are building, to support a collaborative MARS environment that is distributed across both our campuses.

4.1 Acknowledgements

This work was funded in part by Office of Naval Research Contracts N00014-97-1-0838, N00014-99-1-0249, and N00014-99-1-0394; the Advanced Network & Services National Tele-Immersion Initiative; and gifts from Intel, Microsoft, Mitsubishi, and IBM.

References

- [1] R. T. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–385, Aug. 1997.
- [2] H. Beadle, B. Harper, G. Maguire Jr., and J. Judge. Location aware mobile computing. In *Proc. ICT '97 (IEEE/IEE Int. Conf. on Telecomm.)*, Melbourne, Australia, 1997.
- [3] M. Billingham, J. Bowskill, M. Jessop, and J. Morphett. A wearable spatial conferencing space. In *Proc. ISWC '98 (Second Int. Symposium on Wearable Computers)*, pages 76–83, 1998.
- [4] A. Butz, T. Höllerer, S. Feiner, B. MacIntyre, and C. Beshers. Enveloping users and computers in a collaborative 3D augmented reality. In *Proc. IWAR '99 (Int. Workshop on Augmented Reality)*, San Francisco, CA, October 20–21 1999.
- [5] S. Feiner, B. MacIntyre, T. Höllerer, and A. Webster. A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment. In *Proc. ISWC '97 (First Int. Symp. on Wearable Computers)*, pages 74–81, Cambridge, MA, October 13–14 1997.
- [6] S. Feiner, B. MacIntyre, and D. Seligmann. Knowledge-based augmented reality. *Communications of the ACM*, 36(7):52–62, July 1993.

- [7] S. Feiner and A. Shamash. Hybrid user interfaces: Breeding virtually bigger interfaces for physically smaller computers. In *Proc. UIST '91*, pages 9–17. ACM press, 1991.
- [8] T. Höllerer, S. Feiner, and J. Pavlik. Situated documentaries: Embedding multimedia presentations in the real world. In *Proc. ISWC '99 (Third Int. Symp. on Wearable Computers)*, San Francisco, CA, October 18–19 1999.
- [9] B. Jang, J. Kim, H. Kim, and D. Kim. An outdoor augmented reality system for GIS applications. In Y. Ohta and H. Tamura, editors, *Mixed Reality, Merging Real and Virtual Worlds*, pages 391–399. Ohmsha/Springer, Tokyo/New York, 1999.
- [10] S. Long, D. Aust, G. Abowd, and C. Atkenson. Cyberguide: Prototyping context-aware mobile applications. In *CHI '96 Conference Companion*, pages 293–294, April 1996.
- [11] J. Loomis, R. Golledge, and R. Klatzky. Personal guidance system for the visually impaired using GPS, GIS, and VR technologies. In *Proc. Conf. on Virtual Reality and Persons with Disabilities*, Millbrae, CA, June 17–18 1993.
- [12] B. MacIntyre and S. Feiner. Language-level support for exploratory programming of distributed virtual environments. In *Proc. UIST '96*, pages 83–94, Seattle, WA, November 6–8 1996.
- [13] B. MacIntyre and S. Feiner. A distributed 3D graphics library. In *Computer Graphics (Proc. ACM SIGGRAPH '98)*, Annual Conference Series, pages 361–370, Orlando, FL, July 19–24 1998.
- [14] J. Rekimoto and M. Saitoh. Augmented surfaces: A spatially continuous work space for hybrid computing environments. In *Proc. ACM CHI '99*, Pittsburgh, PA, May 15–20 1999. ACM Press.
- [15] A. Smailagic and R. Martin. Metronaut: A wearable computer with sensing and global communication capabilities. In *Proc. ISWC '97 (First Int. Symp. on Wearable Computers)*, pages 116–122, Cambridge, MA, October 13–14 1997.
- [16] J. Spohrer. WorldBoard—What Comes After the WWW? <http://www.worldboard.org/pub/spohrer/wbconcept/default.html>, 1997.
- [17] Z. Szalavari, E. Eckstein, and M. Gervautz. Collaborative gaming in augmented reality. In *Proc. VRST '98*, pages 195–204, Taipei, Taiwan, November 1998.
- [18] Z. Szalavari, D. Schmalstieg, A. Fuhrmann, and M. Gervautz. ”Studierstube”: An environment for collaboration in augmented reality. *Virtual Reality*, 3(1):37–48, 1998.
- [19] B. Thomas, V. Demczuk, W. Piekarski, D. Hepworth, and B. Gunther. A wearable computer system with augmented reality to support terrestrial navigation. In *Proc. ISWC '98 (Second Int. Symp. on Wearable Computers)*, pages 168–171, Pittsburgh, PA, October 19–20 1998.

Situated Documentaries: Embedding Multimedia Presentations in the Real World

Tobias Höllerer
Steven Feiner
Dept. of Computer Science
Columbia University
New York, NY 10027
{htobias,feiner}@cs.columbia.edu

John Pavlik
Center for New Media
Graduate School of Journalism
Columbia University
New York, NY 10027
jp35@columbia.edu

Abstract

We describe an experimental wearable augmented reality system that enables users to experience hypermedia presentations that are integrated with the actual outdoor locations to which they are relevant. Our mobile prototype uses a tracked see-through head-worn display to overlay 3D graphics, imagery, and sound on top of the real world, and presents additional, coordinated material on a hand-held pen computer. We have used these facilities to create several situated documentaries that tell the stories of events that took place on our campus. We describe the software and hardware that underly our prototype system and explain the user interface that we have developed for it.

1. Introduction

Mobile and wearable computing systems provide users access to computational resources even when they are away from the static infrastructure of their offices or homes. One of the most important aspects of these devices is their potential to support *location-aware* or *location-based* computing, offering services and information that are relevant to the user's current locale [1]. Research and commercial location-aware systems have explored the utility of a variety of coarse position-tracking approaches, ranging from monitoring infrared signals emitted by "active badges" [23], to relying on wireless paging cell size to provide local weather and traffic updates [18].

Augmented reality, which demands far more accurate position tracking combined with accurate orientation tracking, can provide an especially powerful user interface for location-aware mobile computing. By supplementing the real world with virtual information, augmented reality can substantially enrich the user's experience of her environ-

ment and present her with an integrated user interface for interacting with the surrounding augmented material.

We have been experimenting with using a mobile augmented reality system (MARS) testbed to create location-aware multimedia presentations for outdoor users. Building on our earlier work on a MARS campus tour guide [7], we introduce the concept of a *situated documentary* that embeds a narrated multimedia documentary within the same physical environment as the events and sites that the documentary describes. One of the most important principles of journalism is to locate a story in a physical space. We accomplish this by situating the news consumer literally at the story's location, and layering a multimedia documentary over that space.

As depicted in Figure 1, the user wears an experimental backpack-based system, based on commercial hardware that we have chosen for programmability and power at the expense of comfort and wearability. Graphics and imagery are overlaid on the surrounding world by a see-through head-worn display. Head tracking is accomplished using a centimeter-level real-time kinematic GPS position tracker and an inertial/magnetometer orientation tracker. Audio is presented through the head-worn display's earphones, and coordinated video and other multimedia material are presented on a companion hand-held display. Interaction occurs through a set of selection mechanisms based on positional proximity and gaze orientation, a trackpad that is used with the head-worn display, and a pen-based user interface on the hand-held display.

In this paper, we first discuss how our work relates to previous research in Section 2. Next, in Section 3, we introduce our main application scenario and its user interface techniques: a multimedia documentary of highlights in the history of Columbia's campus. We then briefly describe the hardware and software used for our current testbed in Sec-

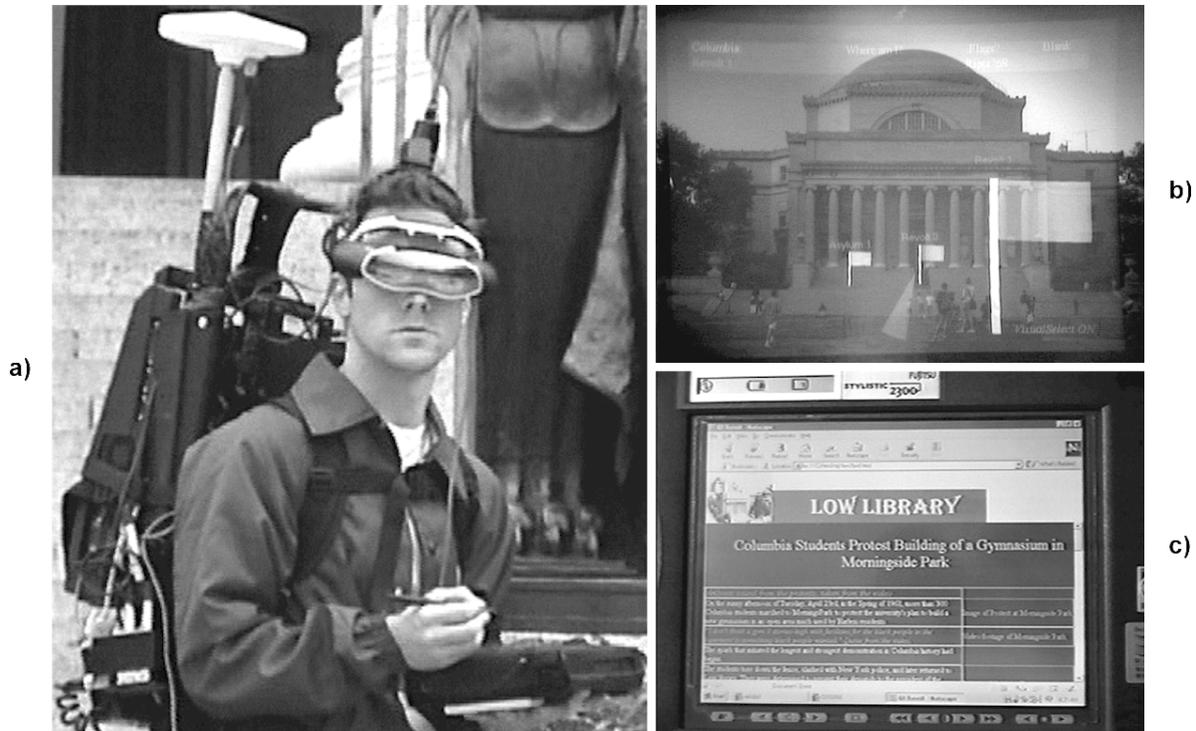


Figure 1. Situated documentaries. a) Our backpack-based testbed, with tracked see-through head-worn display and pen-based hand-held computer. b) An image photographed by a video camera that wears our testbed's see-through head-worn display. The labels and virtual flags are part of the user interface, described in Section 3. c) Related information displayed on our hand-held computer.

tion 4. Finally, Section 5 provides our conclusions and a discussion of ongoing and future work.

2. Related Work

As computers continue to shrink in size, researchers have begun to address the development of outdoor location-aware mobile and wearable systems. Some have relied on modifying the environment being explored; for example, Smailagic and Martin [19] label campus information signs with bar codes to provide location-specific information on a hand-held computer equipped with a bar code scanner. In contrast, others have combined GPS and orientation trackers to produce map-based contextual displays [11], to provide audio navigation assistance to blind users [12], and to annotate the world with overlaid textual labels [7, 22, 9].

Situated documentaries rely in part on the idea of creating hypertextual links between physical and virtual objects or locations. In earlier indoor work, using short-range, magnetic and ultrasonic tracking systems, we developed a hypermedia system that supports linking arbitrary X11 windows, displayed on a tracked see-through head-worn dis-

play, to a variety of targets, including 3D world locations and tracked objects [6]. Wearable systems by Rekimoto et al. [17] and Starner et al. [21] allow people to register digital data with visually-coded or infrared-tagged objects. Billingham et al. [2] use similar visual fiducials to position texture-mapped representations of participants in an augmented reality teleconference. Mann [15] and Jebara et al. [10] associate information with untagged objects using visual recognition algorithms. Pascoe [16] uses a hand-held display and GPS to allow an ecologist to link observation notes to the locations at which they are written. All these projects can be seen as leading towards the goal articulated in Spohrer's proposal for a "WorldBoard" [20]: the creation of a world-wide spatial hypertext of information anchored to physical locations and objects.

Our work is built on top of a new version of the backpack-based wearable MARS testbed that we developed for our earlier "Touring Machine" [7]. This system uses a campus database to overlay labels on buildings seen through a tracked head-worn display. Users can request additional overlaid information, such as the names of a building's departments, and can view related information, such as a de-



Figure 2. Virtual flags denoting points of interest, photographed from the top of a campus building.

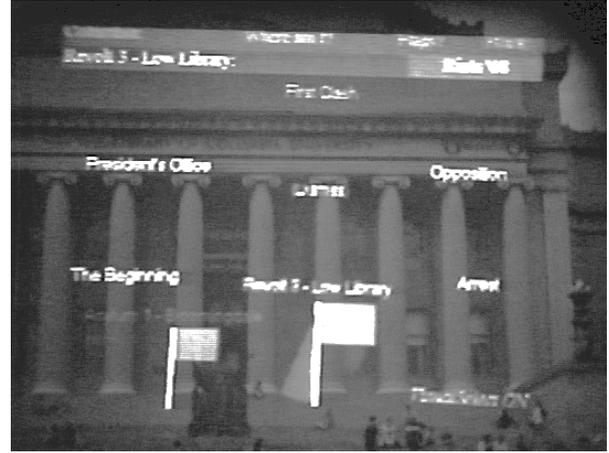
partment’s web page, on a hand-held display. The situated documentaries that we describe here extend this previous work in several ways:

- Rather than linking individual labels or web pages to locations, we support context-dependent, narrated multimedia presentations that combine audio, still images, video, 3D graphics, and omnidirectional camera imagery.
- We make extensive use of overlaid 3D graphics for both the user interface (e.g., 3D widgets for user guidance) and the presentation content (e.g., *in situ* reconstructions of buildings that no longer exist and views of visually obstructed infrastructure).
- We embed the informational elements in an early version of a new *physical hypermedia* user interface that guides users through a presentation, while giving them the freedom to follow their own trails through the material.

3. User Interface

Our user stands in the middle of Columbia’s campus, wearing our experimental backpack computer system and a see-through head-worn display, and holding a tablet computer (Figure 1a). As the user moves about, their position and head orientation are tracked, and through the head-worn display they see the campus environment overlaid with virtual material, such as that shown in Figures 1(b) and 2.

The user can interact with the surrounding environment in different ways. On the hand-held computer, which is net-



a)



b)

Figure 3. Two different menu designs for listing multimedia snippets about the student revolt. a) World-stabilized circular menu around Low Library (photographed through an earlier, low-resolution, see-through, head-worn display). b) Head-stabilized list with anchor to its flag (screen dump of the system running in indoor test mode, with an omnidirectional image as a backdrop).

worked to the backpack computer that drives the head-worn display, the user can view and interact with information, and input data with a stylus. All information on the hand-held display is presented using a standard web browser. Items seen on the head-worn display can be selected with an approximation to gaze-oriented selection described below. A menu on the head-worn display can be manipulated using a two-button trackpad mounted on the back of the hand-held computer for easy “reach-around” selection.

The head-worn user interface consists of a screen-

stabilized part and a world-stabilized part. The menu bars on top of the screen and the cone-shaped pointer at the bottom (shown most clearly in Figure 3b) are screen-stabilized and therefore always visible. World-stabilized material is visually registered with specific locations on campus. World-stabilized 3D elements are displayed in the correct perspective for the user's viewpoint, so the user can walk up to these elements just as they can to physical objects

3.1 Application Scenario

Our situated documentary begins with a narrated introduction, explaining that the user will be able to learn about events related to the campus, and referring the user to the hand-held display for an overview. Before turning to the hand-held computer, the user looks around and sees virtual flags with textual labels denoting points of interest, positioned around the campus (see Figures 1b and 2). The virtual flags are world-stabilized user-interface elements that are iconic representations of the topmost *group nodes* in a hierarchical presentation.

The hand-held display provides an overview of the material embedded in the surrounding environment. Three main topics are currently available: a description of the Bloomingdale Asylum for the Insane, which once occupied the current campus before Columbia's move in the late 19th century, a documentary on the Columbia student revolt of 1968, and a tour of Columbia's extensive underground tunnel system. Looking at the surrounding flags, the user can see how the different stories are distributed over the campus area. The labeled flags come in three different colors: red for the student revolt, blue for the tunnel system, and green for the Bloomingdale Asylum.

The user can select a flag in several different ways. One method, which works when the user is in the system's *VisualSelect* mode, is to look in the flag's direction, orienting one's head so the desired flag's projection is closer than any other to the center of the head-worn display and within a fixed target area. When these criteria are met, the flag's label changes color to yellow. If the criteria hold for a half second, then the flag is selected and its label changes color to green. (This approximation of gaze selection was originally developed for selection of building tags in [7].) Flags are selectable from any distance. Although the flags scale with distance, their textual labels do not, so there is always a visible anchor that is selectable.

A second selection method is based on positional proximity. A menu item allows the user to ask the system to select the flag to which they are currently closest (or to select another flag by name), and the cone-shaped pointer on the head-worn display will point towards that flag, guiding the user to it. Finally, a flag can be selected automatically by following a link in the presentation.

When a flag is selected, it starts to wave gently, and all flags of a different color are dimmed (reduced in intensity). Therefore, when a user looks around while a flag is selected, the other flags in its category stand out. The cone-shaped pointer always points toward the selected flag, so that the user can be guided back to it should they look away.

Selecting a flag causes the second menu bar (the green *context* menu below the blue top-level menu) to display that flag's label plus additional entries that are available for its group node (e.g., links to other group nodes). All these entries can be selected using the trackpad. The group nodes (and their corresponding flags) have a default numbering corresponding to an order set forth in the presentation description. A button click on the trackpad directs the user to the next node in this order; however, at all times the user can choose to select a different flag using any of the methods mentioned above.

In our case, the user selects the entry for the student revolt from the overview menu on the hand-held computer. The cone-shaped arrow on the head-worn display points to a red flag, which starts waving, in front of Low Library, which is about 150 yards away. This flag is the starting point for information on the student revolt.

Once a flag is selected, the user can display an overlaid *in-place* menu (see Figure 3), which lists the parts of the presentation associated with the flag's group node. (Section 3.3 discusses the in-place menus further.) The in-place menu for Low Library's revolt flag provides access to background information on how the student revolt started, grouped into five segments.

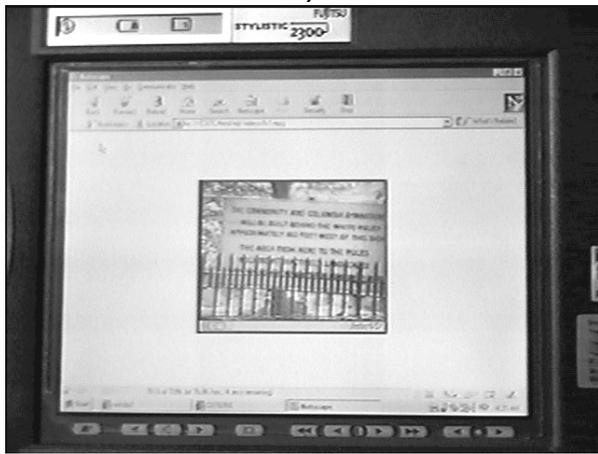
Selecting an entry in this menu using the trackpad starts that entry's part of the multimedia presentation, each of which ranges in length from seconds to minutes in our current material. Here, the user selects the entry labeled *First Clash*. This results in a narrated description of how the students and the police clashed for the first time on the steps of Low Library, where the user is now looking. The presentation includes coordinated still images that are overlaid on the scene (Figure 4a) and videos that are played on the hand-held computer (Figure 4b).

The head-worn display's menu bar allows the user to display an overview of the student revolt on the hand-held computer or to follow links to other places directly by selecting them with the trackpad to learn more about about the revolt and what happened at other campus buildings.

At this point, the user has found a description of how the students used Columbia's tunnel system to occupy buildings guarded aboveground by the police. The user decides to follow a link to learn more about the tunnels by exploring the blue flags. Since the real tunnels are difficult (and illegal) to enter, the user can vicariously explore portions of them through a set of 360° omnidirectional camera photo-



a)

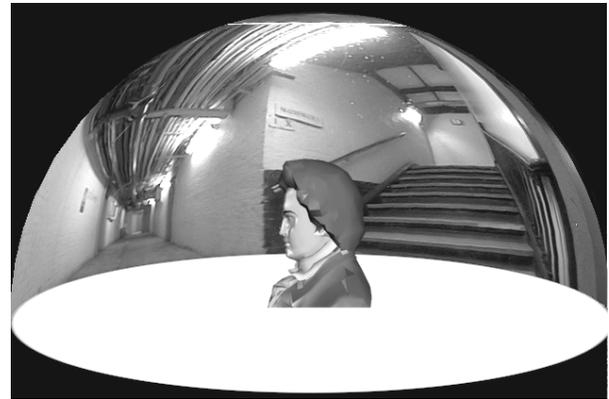


b)

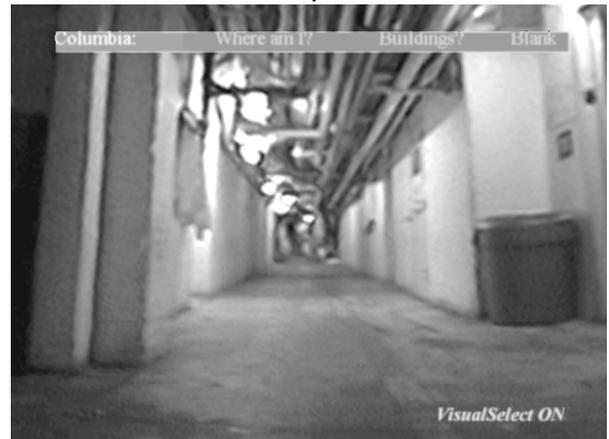
Figure 4. Imagery documenting the student revolt in 1968: a) Still image, overlaid on top of Low Library, b) video material displayed on the hand-held computer

graphic images (Figure 5) that temporarily teleport the user underground, supplemented by maps and blueprints.

The presentation mentions that the oldest parts of the tunnel system preceded Columbia's move to the area and were originally built for the Bloomingdale Asylum. Intrigued, our user turns to the green flags to find out where the main asylum buildings were situated, and is shown a 3D model of the buildings overlaid in place on the campus, in conjunction with historical images (see Figure 6). The documentary mentions that one building built for the asylum is still standing and is now known as Buell Hall, and points the user toward it.



a)



b)

Figure 5. Exploring Columbia's tunnel system: a) Schematic view of how a user experiences an omnidirectional camera image. b) The omnidirectional camera image seen from a user's perspective.

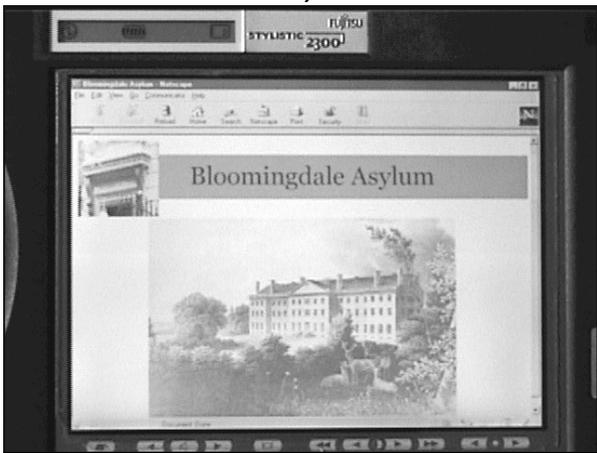
3.2. Multimedia Presentations

The multimedia material in each presentation node is a coordinated media stream (see Section 4.2) that typically, but not necessarily, makes use of both the hand-held display and the head-worn display, and which includes an audio track. The different media that can be freely combined to create a multimedia presentation are:

- *Audio material on the head-worn display.* Audio is played over the head-worn display's earphones, and includes both narration and non-speech audio (e.g., recordings of the 1968 revolt).
- *Images on the head-worn display.* Images (e.g., Figure 4a) are displayed as world- or head-stabilized 3D textured polygons that can make use of simple ani-



a)



b)

Figure 6. a) A simplified 3D model of the main Bloomingdale asylum building overlaid on Columbia's campus by the see-through head-worn display. b) Documentary material displayed on the hand-held computer.

mated effects. For example, we often “flip up” head-stabilized images from a horizontal position until they fill the screen.

- *Web pages that include static images, video material, and applets on the hand-held display.* Figures 4(b) and 6(b) show examples of images and video, created by calling up related material on the hand-held browser using our communication infrastructure (see Section 4.2).
- *3D models.* Figure 6(a) shows a simple example. Models are shown full-size and world-stabilized in their actual location.

- *360° omnidirectional camera surround views.* These allow us to immerse the user in an environment that is not physically available. We use a commercial omnidirectional camera [5]: a digital camera pointing at a parabolic mirror that captures a 360° hemispherical surround view in a single image. Each of these anamorphic images is texture-mapped onto a hemisphere displayed around the user, as depicted schematically in Figure 5(a), so that the user can look around (Figure 5b). The see-through head-worn display's opacity is controlled by a dial, allowing us to make the display opaque when viewing these images. (Unfortunately, the display's opacity cannot be set in software.)

3.3. Exploratory UI Design

We also use omnidirectional images as backdrops for indoor demonstrations of our system and for exploratory development of new user interface elements and variants. Figure 3 demonstrates this approach. Part (a) shows our original version of an in-place menu, shot outdoors through a low-resolution see-through head-worn display; part (b) shows our current version of the same menu, captured as a screen dump of the system running indoors, using an omnidirectional image of the campus as a backdrop. In the latter design, the menu is a head-stabilized element, rather than the world-stabilized circular menu of part (a). A leader line links the menu to its associated flag, allowing it to be followed back if the user turns away from the flag, an approach that we used to direct users to objects that were not within their field of view in an earlier indoor augmented reality system for maintenance and repair [8].

4. System Design

4.1. Hardware

Our current backpack is an updated version of our first outdoor MARS testbed [7], with the following changes:

Head-worn Display: We use a Sony LDI-100B color display with 800×600 triad resolution. It has a dial to adjust its opacity from nearly totally opaque to about 20% transparent. In our experience, under a bright cloudy sky the preferred setting is close to the most opaque. We have just begun to experiment with a stereo version of this display, the Sony LDI-D100B.

The images in this paper were shot directly through the LDI-100B display worn by a dummy head containing an embedded NTSC camera. Images 3a) and 4a) stem from earlier footage, shot through a Virtual I/O i-glasses display with 263×230 triad resolution.

Hand-held Computer: The hand-held computer shown in Figures 1, 4(b), and 6(b) is a Fujitsu Stylistic 2300 with

a 233 MHz Pentium MMX CPU and a transfective 800×600 color display, designed to be readable in bright sunlight. The Fujitsu's performance is adequate for playing MPEG movies of up to VGA resolution at reasonable frame rates, but it is heavier than we would like (3.9 pounds). We have just switched to a 2.2 pound Mitsubishi AmITY CP pen-based computer with a 166 MHz Pentium MMX CPU and 640×480 color display.

Orientation Tracker: We use an Intersense IS-300Pro inertial/magnetometer orientation tracker with a single sensor mounted rigidly on a head band that we attached to the head-worn display's temple pieces, as shown in Figure 1(a).

Position Tracker: Position tracking is done with an Ashtech GG24 Surveyor real-time kinematic differential GPS system, which uses both US GPS and Russian Glonass satellite constellations to increase the number of visible satellites. We have installed a base station on campus, from which we broadcast correction signals via radio modem. This system provides centimeter-level accuracy in open areas, such as those depicted in the figures, where we have line-of-sight to more than six satellites. However, tracking degradation and loss remain a problem when we pass too close to tall buildings or beneath trees.

4.2. Software

We extended the software architecture of our previous prototype [7], which is based on our COTERIE distributed virtual environment infrastructure [13, 14]. We run a custom-built HTTP server on the hand-held computer, allowing it to communicate with the backpack computer and accept user input from any web-based interface, including Java applets.

The multimedia information to be conveyed through the augmented reality interface has to be arranged and locally distributed over the target region. For this purpose we designed several authoring tools.

To create the multimedia presentations, we developed a simple extension to the interpreted language *Repo*, our extended variant of the lexically scoped interpreted language *Obliq* [4]. Each multimedia presentation is stored as a *Repo* script, referencing by filename the multimedia "chunks" (images, video segments, audio snippets, 3D animations, omnidirectional views) it uses. Each chunk is stored on the computer (backpack or hand-held) on which it is to be played; additional material to be presented on the hand-held computer can be obtained from the web using a wireless network interface.

Students in a graduate Journalism class taught by the third author used our multimedia prototyping environment to break the footage they had collected into chunks and wrote scripts to create our multimedia presentations. Syn-

chronization takes place purely at the level of these relatively coarse-grain media chunks by exchanging *Repo* messages between the main server on the backpack computer and the HTTP server on the hand-held computer.

All location-based information is stored in a campus database on the backpack computer. This database contains the complete structure of the situated documentaries, including the contents of all context-menus and links to the multimedia presentation scripts.

We used an early version of a map-based tool we are developing to place 3D objects at any specified latitude-longitude. For this project, we scanned in a high-resolution map of Columbia's campus that provides a placement resolution of about 6 inches in latitude or longitude.

5. Conclusions and Future Work

Although most of our user experience has been limited to the authors of this paper and to the students who helped construct the presentations, our system has been demonstrated informally in several Journalism classes, to visitors to our lab, and to attendees of a Department of Defense seminar who tried the indoor version. While feedback has been encouraging, users understandably cite the current prototype's form factor, weight (about forty pounds), and appearance as drawbacks. We are confident, however, that these issues will be addressed by the commercial development of sufficiently small wearable devices.

For the near term, we note that much of our backpack's weight is due to its computer, which together with its external battery weighs about twenty-two pounds. We selected this machine (Fieldworks 7600) for the programming comfort of the system's developers, rather than the physical comfort of its wearers. Its flexibility and extensibility (expansion ports for six PCI and ISA cards, and the ability to run a desktop operating system and programming environment) have been invaluable during development and testing. We are investigating options for replacing it with a lighter, more powerful laptop, but require high-performance support for the OpenGL 3D graphics API that we use, which is not yet offered by current laptops. To provide a lighter hand-held display, we are beginning to experiment with the Casio Cassiopeia E-100 running Windows CE, a palm-top computer with a 240×320 16-bit color display.

There are many directions that we are currently exploring to further develop our software. For example, our system currently provides no reasonable facilities for end-user authoring. We are especially interested in developing this kind of support, with emphasis on how such a system might be used by journalists in the field to develop stories. We are also working on an interface between our backpack system and an indoor multi-user augmented reality system [3]

to make possible collaboration among indoor and outdoor users. Using a 3D model of the environment, indoor users create virtual objects and highlight real objects for outdoor users to see, and maintain histories of outdoor users' activities. In turn, outdoor users point out interesting objects and events for indoor users to view.

6. Acknowledgements

We thank Blair MacIntyre for developing Coterie, for his work on the first generation MARS testbed on which this work builds, and for his advice on Coterie programming issues. Gus Rashid developed the map-based tool mentioned in Section 4.2, and Elias Gagas helped write the GPS drivers and assisted in testing and improving the system. Students in John Pavlik's Journalism courses in the Center for New Media collected multimedia material, turned it into coherent presentations, and participated in helpful discussions on the user interface. In particular we would like to thank Aklilu Hailemariam, Tali Dayan, Dave Westreich, Stephen Newman, Dave Terraso, Dave Derryck, and Sheryl LeDuc.

This work was supported in part by Office of Naval Research Contracts N00014-97-1-0838, N00014-99-1-0249, and N00014-99-1-0394; and gifts from IBM, Intel, Microsoft, and Mitsubishi.

References

- [1] H. Beadle, B. Harper, G. Maguire Jr., and J. Judge. Location aware mobile computing. In *Proc. ICT '97 (IEEE/IEE Int. Conf. on Telecomm.)*, Melbourne, Australia, 1997.
- [2] M. Billinghurst, J. Bowskill, M. Jessop, and J. Morphet. A wearable spatial conferencing space. In *Proc. ISWC '98 (Second Int. Symposium on Wearable Computers)*, pages 76–83, 1998.
- [3] A. Butz, T. Höllerer, S. Feiner, B. MacIntyre, and C. Beshers. Enveloping users and computers in a collaborative 3D augmented reality. In *Proc. IWAR '99 (Int. Workshop on Augmented Reality)*, San Francisco, CA, October 20–21 1999.
- [4] L. Cardelli. A language with distributed scope. *Computing Systems*, 8(1):27–59, Jan 1995.
- [5] Cyclovision Technologies, Inc. ParaShot—One Shot 360 Degree Surround View Images. <http://www.cyclovision.com>, 1998.
- [6] S. Feiner, B. MacIntyre, M. Haupt, and E. Solomon. Windows on the world: 2D windows for 3D augmented reality. In *Proc. UIST '93*, pages 145–155, 1993.
- [7] S. Feiner, B. MacIntyre, T. Höllerer, and A. Webster. A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment. In *Proc. ISWC '97 (First Int. Symp. on Wearable Computers)*, pages 74–81, Cambridge, MA, October 13–14 1997.
- [8] S. Feiner, B. MacIntyre, and D. Seligmann. Knowledge-based augmented reality. *Communications of the ACM*, 36(7):52–62, July 1993.
- [9] B. Jang, J. Kim, H. Kim, and D. Kim. An outdoor augmented reality system for GIS applications. In Y. Ohta and H. Tamura, editors, *Mixed Reality, Merging Real and Virtual Worlds*, pages 391–399. Ohmsha/Springer, Tokyo/New York, 1999.
- [10] T. Jebara, B. Schiele, N. Oliver, and A. Pentland. DyPERS: Dynamic personal enhanced reality system. In *Proc. 1998 Image Understanding Workshop*, Monterey, CA, November 1998.
- [11] S. Long, D. Aust, G. Abowd, and C. Atkinson. Cyberguide: Prototyping context-aware mobile applications. In *CHI '96 Conference Companion*, pages 293–294, April 1996.
- [12] J. Loomis, R. Golledge, and R. Klatzky. Personal guidance system for the visually impaired using GPS, GIS, and VR technologies. In *Proc. Conf. on Virtual Reality and Persons with Disabilities*, Millbrae, CA, June 17–18 1993.
- [13] B. MacIntyre and S. Feiner. Language-level support for exploratory programming of distributed virtual environments. In *Proc. UIST '96*, pages 83–94, Seattle, WA, November 6–8 1996.
- [14] B. MacIntyre and S. Feiner. A distributed 3D graphics library. In *Computer Graphics (Proc. ACM SIGGRAPH '98)*, Annual Conference Series, pages 361–370, Orlando, FL, July 19–24 1998.
- [15] S. Mann. Wearable computing: A first step toward personal imaging. *IEEE Computer*, 30(2), February 1997.
- [16] J. Pascoe. Adding generic contextual capabilities to wearable computers. In *Proc. ISWC '98 (Second Int. Symp. on Wearable Computers)*, pages 92–99, Cambridge, MA, October 19–20 1998.
- [17] J. Rekimoto, Y. Ayatsuka, and K. Hayashi. Augment-able reality: Situated communication through physical and digital spaces. In *Proc. ISWC '98 (Second Int. Symp. on Wearable Computers)*, pages 68–75, Cambridge, MA, October 19–20 1998.
- [18] Seiko Communications. Seiko message watch documentation. <http://www.messagewatch.com>, 1998.
- [19] A. Smailagic and R. Martin. Metronaut: A wearable computer with sensing and global communication capabilities. In *Proc. ISWC '97 (First Int. Symp. on Wearable Computers)*, pages 116–122, Cambridge, MA, October 13–14 1997.
- [20] J. Spohrer. WorldBoard—What Comes After the WWW? <http://www.worldboard.org/pub/spohrer/wbconcept/default.html>, 1997.
- [21] T. Starner, S. Mann, B. Rhodes, J. Levine, J. Healey, D. Kirsch, R. Picard, and A. Pentland. Augmented reality through wearable computing. *Presence*, 6(4):386–398, August 1997.
- [22] B. Thomas, V. Demczuk, W. Piekarski, D. Hepworth, and B. Gunther. A wearable computer system with augmented reality to support terrestrial navigation. In *Proc. ISWC '98 (Second Int. Symp. on Wearable Computers)*, pages 168–171, Pittsburgh, PA, October 19–20 1998.
- [23] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. *ACM Trans. on Information Systems*, 10(1):91–102, January 1992.