THE UNIVERSITY OF CALGARY

Multimodal Co-located Interaction

by

Edward Hiatt Tse

A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE

DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

CALGARY, ALBERTA

DECEMBER 2007

THE UNIVERSITY OF CALGARY

FACULTY OF GRADUATE STUDIES

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled "Multimodal Co-located Interaction" submitted by Edward Hiatt Tse in partial fulfillment of the requirements for the degree Doctor of Philosophy.

[illegible]

Supervisor, Saul Greenberg
Department of Computer Science

[illegible]

Sheelagh Carpendale
Department of Computer Science

Carey Williamson

Carey Williamson
Department of Computer Science

[illegible]

Chia Shen
Mitsubishi Electric Research Laboratories

[illegible]

Patrick Feng
Faculty of Communications and Culture

[illegible]

Phil Cohen
Adapx

November 6, 2007

Date

Abstract

With the advent of very large, high resolution and affordable digital displays researchers are investigating interaction techniques suitable for collaborative work. This dissertation explores the design and technical development of technologies that support multiple co-located people collaborating with multimodal speech and gesture commands over digital table. In this thesis I first explain why it would be useful to use speech and gesture commands in a co-located setting through a set of behavioural foundations summarizing theories, empirical and ethnographic research on how people collaborate in everyday settings. I then described how to design multi user speech and gesture interfaces atop of existing single user applications. I then observed how people use these wrappers for collaborative work and found that people overwhelmingly used speech and gestures as both commands to the computer and as communication to other collaborators. To examine the effectiveness of speech filtering for selection over a large digital wall display I performed a controlled evaluation and found that multimodal commands improved selection efficiency, accuracy and user preference over two gesture-only selection techniques.

The remainder of this thesis described further technical explorations in the design of multimodal co-located applications. I explained how to create multimodal co-located systems using GSI DEMO to map speech and gesture commands to keyboard and mouse actions by demonstration rather than programming. For example, continuous gestures were trained by saying "Computer, when I do [one finger gesture], you do [mouse drag]".

I supported parallel work over existing applications using a multimodal split view tabletop, a tabletop whose surface is split into two adjacent projected views connected to separate computers. By using separate computers pairs could work in parallel over existing single user application over a shared tabletop surface. Multimodal commands provided enhanced activity awareness especially when people were working in parallel.

Finally, I explored how multimodal co-located interaction could be applied to a collaboration-aware groupware system for supporting the brainstorming activities of industrial designers. I presented several design issues that arose in the multimodal co-located groupware setting.

Publications

Materials, ideas, and figures from this thesis have appeared previously in the following publications:

Journal Articles

Tse, E., Greenberg, S., Shen, C. and Forlines, C. (2007) Multimodal Multiplayer Tabletop Gaming. In ACM CIE Computers in Entertainment. June. ACM Press (this is a reprint rewarded to the best papers at Pervasive Games)

Papers

Tse, E., Shen, C., Greenberg, S. and Forlines, C. (2006) Enabling Interaction with Single User Applications through Speech and Gestures on a Multi-User Tabletop. Proceedings of Advanced Visual Interfaces (AVI'06), May 23-26, 336-343, Venezia, Italy, ACM Press.

Tse, E., Greenberg, S., Shen, C. and Forlines, C. (2006) Multimodal Multiplayer Tabletop Gaming. Proceedings Third International Workshop on Pervasive Gaming Applications (PerGames'06), in conjunction with 4th Intl. Conference on Pervasive Computing, (May 7th Dublin, Ireland), 139-148.

Tse, E., Greenberg, S. and Shen, C. (2006) GSI DEMO: Multi User Gesture / Speech Interaction over Digital Tables by Wrapping Single User Applications. Proc Eighth International Conference on Multimodal Interfaces (ICMI'06), (Nov 2-4, Banff, Canada), ACM Press, 76-83.

Tse, E., Shen, C., Greenberg, S. and Forlines, C. (2007) How Pairs Interact Over a Multimodal Digital Table. Proceedings of ACM CHI Conference on Human Factors in Computing Systems (May 1-3, San Jose, CA), ACM Press, 215-218.

Tse, E., Hancock, M. and Greenberg, S. (2007) Speech-Filtered Bubble Ray: Improving Target Acquisition on Display Walls. Proceedings of ICMI 2007 (Nov 12-15, Nagoya, Japan), 307-314.

Tse, E., Greenberg, S., Shen, C., Barnwell, J., Shipman, S. and Leigh, D. (2007) Multimodal Split View Tabletop Interaction Over Existing Applications. Proceedings of IEEE Tabletop 2007 (Oct 10-12, Newport, RI), 129-136.

Tse, E., Greenberg, S., Shen, C., Forlines, C., and Kodama, R. (2007) Exploring True Multi-User Multimodal Interaction over a Digital Table, Proceedings of ACM DIS Conference (Feb 25-27, Cape Town, South Africa), In Press.

Demos, Videos and Posters

Tse, E., Greenberg, S. and Shen, C. (2006) Motivating Multimodal Interaction Around Digital Tabletops. Video Proceedings of ACM CSCW'06 Conference on Computer Supported Cooperative Work, November, ACM Press.

Tse, E., Greenberg, S., Shen, C. (2006) Multi User Multimodal Tabletop Interaction over Existing Single User Applications. Demo, Adjunct Proc ACM CSCW 2006.

Tse, E., Greenberg, S., Shen, C. (2006) Exploring Interaction with Multi User Speech and Whole Handed Gestures on a Digital Table. Demonstration, Adjunct Proc ACM UIST 2006.

Tse, E. (2006) Multimodal Co-located Collaboration. Doctorial Consortium, Adjunct Proc. ACM UIST 2006.

Selected General Media Coverage

"…this new DiamondTouch touch panel table interface has one thing NYU's unit lacked: game… Mitsubishi's system also includes voice control, and they show it all off with a fairly impressive demo of WarCraft III"

Miller, P. (2006) Mitsubishi R&D rocks Warcraft III with new DiamondTouch table, Engadget (Mar 27, 2006), http://www.engadget.com/2006/03/27/mitsubishi-randd-rocks-warcraft-iii-with-new-diamondtouch-table.

"Mitsubishi Electric Research Laboratories gave us these videos of Google Earth in use. Word is the military got some demos and is pretty interested in the technology. It's pretty easy to figure out why if you watch the vids."

Block, G. (2006) Google Earth on the DiamondTouch, International Gaming News (Mar 24, 2006), http://gear.ign.com/articles/698/698449p1.html.

"It changes everything and nothing: it's the same game, just far more controllable and intuitive."

Pearson, C. (2006) Digital Input: Wanted touchscreen gaming, PC Gamer (June, 2006), Issue 162, page 15.

"The player, pretending to be a commander, can control movement by tapping the display and shouting commands"

Kageyama, Y. (2006) Japanese Touch Panel Table. Associated Press, Yahoo News and CNN (June 1, 2006), http://news.yahoo.com/s/ap/20060601/ap_on_hi_te/japan_touch_panel_table_1

"Forget 24-inch monitors, get yourself a big table. This, quite literally, is brief glimpse into the future"

Sbarski, P., (2006) Forget 24-inch monitors, get yourself a big table, The Inquirer (June 14, 2006), http://www.theinquirer.net/?article=32422.

"Expect a Google job offer to be coming, Edward. Expect that tabletop display to be showing up in the Googleplex, which has been woefully lacking in cool stuff for visitors lately."

Sullivan, D. (2006) Minority Report-Like Interactive Google Earth, SearchEngineWatch.com (June 14, 2006), http://blog.searchenginewatch.com/blog/060614-065502.

Acknowledgments

"What cannot be achieved in one lifetime will happen when one lifetime is joined to another" – Harold Kushner

This work would not have been possible without the support and guidance of many people and organizations. This section acknowledges the fact this work is a small part of a much larger community of friends, family, mentors, colleagues and fellow researchers.

To Saul my supervisor, mentor, and friend, thank you for your support, guidance and direction during my career as a graduate student. You took me in as an undergraduate researcher and have taught me the skills I need to develop and disseminate ideas. You've taught me the skills of communicating novelty through writing, presentations and demonstrations. Thank you for teaching me to focus my creativity and to critically evaluate my own ideas. Your guidance has given me the wisdom to see the big picture of my own work. My graduate career has been a long process of growth. Thank you for being patient and kind, and for stretching your research boundaries so that I could explore Multimodal Co-located Interaction. Thank you for providing opportunities to connect and collaborate with leading researchers in this field. You have always gone above and beyond the call of duty - celebrating the accomplishments of your students and providing critical feedback when necessary. Words cannot adequately express the gratitude I feel towards your contributions to my personal growth.

To Chia my supervisor at Mitsubishi Electric Research Laboratories, thank you for accepting me as an intern during the early stages of my PhD. This opportunity connected me with a world-class community of researchers and taught me about invention and innovation in an industry setting. Thank you for your forgiveness and support with my weekend experiment with Warcraft III that ultimately became the basis for this PhD dissertation. You went out of your way to make it possible for me to continue my doctoral dissertation at the University of Calgary, and provided me with all the resources I needed to complete my doctoral dissertation. Thank you for providing me with the opportunities to demonstrate my work at prestigious venues such the New York Police Department and Wired NextFest. I am forever grateful for your guidance and support.

To the members of the Interactions Laboratory at the University of Calgary, I am thankful to have been a part of such an amazing community. This work would not have been possible without your support and input. I am inspired by the passion each of you have shown in your work and your concern for the well being of others. Thank you for all of the wonderful social activities that we have shared together. In addition to my supervisor, special thanks goes to Dane Bertram, Sheelagh Carpendale, Rob Diaz-Marino, Elena Fanea, Cheng Guo, Jens Grubert, Mark Hancock, Uta Hinrichs, Jonathan Histon, Petra Isenberg, Tobias Isenberg, Jeroen Keijser, Russell Kruger, Gregor McEwan, Carman Neustaedter, Natalia Romero, Kathryn Rounding, Mike Rounding, Stacey Scott, Stephanie Smale, Charlotte Tang, Tony Tang, Annie Tat, Kim Tee, Cody Watts, Nelson Wong, Min Xin, and Jim Young. Thanks for your collaborations, friendships, helpful advice and support. Your contributions will not be forgotten.

A great number of people need to be thanked at Mitsubishi Electric Research Laboraties (MERL) for all of their support with my PhD research. In addition to my supervisor special thanks goes to John Barnwell, Octav Chipara, Paul Dietz, Alan Esenther, Clifton Forlines, Bret Harsham, Yuri Ivanov, Johnny Chung Lee, Darren Leigh, Joe Marks, Patrick Pletscher, Kathy Ryall, Sam Shipman, Jay Summit, Jonathan Westhues, Daniel Wigdor, Chris Wren, and William Yerazunis. Thank you for your discussions and support during my internships at MERL.

To Mike Boyle, thank you for providing me with the technical foundations that have helped me turn imagination into reality. You taught me the art of programming and solving technical problems and this skill provided me with the ability to turn paper sketches into real prototypes. Thank you for your friendship and our continued connection throughout my PhD.

To Sheelagh Carpendale, Peggy Chan, Lan Guo, Carman Neustaedter, Chia Shen, and Carey Williamson, thank you for your thoughtful and insightful revisions of this dissertation. Your unique insights into this work have considerably improved the quality of this thesis.

To Gerald Morrison, thank you for your encouragement and support of my PhD work at conferences and within Smart Technologies. Your passion for research collaborations and innovation is inspirational. I look at our future collaborations together with great anticipation.

To my parents Jim and Jane Tse, thank you for your patience and support throughout my academic career. You have guided me through many difficult career decisions and have supported me through all of the difficult and joyful moments. Thank you for always being there for me when I needed you, for celebrating my successes, and for lovingly pointing out my weaknesses.

To my siblings Angela and David Tse, thank you for celebrating my successes and reminding me to keep ego in check. Thank you for being there when I really needed you. I treasure the time we've spent traveling abroad and look forward to our future excursions.

To my grandmother Chun Sim Yan, thank you for all of your dedicated support throughout my life. Thank you for enthusiastically celebrating the new and old phases of my life. Your loving kindness will be forever remembered.

To Shawn Cain my mentor and friend, thank you for strengthening our relationship throughout my PhD. Thank you for encouraging me through your enthusiastic testimonies. Your enthusiasm and passion has inspired me and is reflected in how I talk to others about my work and my journey.

To Simon Chu, thank you for encouraging me to develop my passion for music. I am grateful for the encouragement and testimony you have provided throughout my PhD. Your training, discipline, and passion have been an inspiration for my own work.

To the members of Sharon Lutheran Church and the Calgary Chinese Oratorio Society, thank you for your passion and enthusiasm. Your words of encouragement have been truly inspirational.

This research would not be possible without the generous contributions of our sponsors: The National Sciences and Engineering Research Council (NSERC), the Alberta Ingenuity Fund (AIF), and the Informatics Circle of Research Excellence (iCORE). They have given me the opportunity to explore innovative ideas in the area of large display interaction without the financial risks associated with starting my own business. The learning opportunities that I have received will no doubt be invaluable in all my future endeavours.

Dedication

I dedicate this dissertation to Peggy Yee Pui Chan.

For all the times you stood by me even when we were far apart, rejoiced in my strengths, lovingly reminded me of my weaknesses, went out of your way to help me in times of need, laughed with me, cried with me, rearranged your schedule for me, travelled with me, listened to me, and forgave me when I was wrong, this thesis is dedicated to you.

Table of Contents

List of Tables

List of Figures

Chapter 1. Introduction

This dissertation explores the design and technical development of technologies that support multiple ***co-located*** people collaborating with ***multimodal*** speech and gesture commands over digital table. Within the broad area of multimodal interaction, I concentrate specifically on ***publicly perceptible*** hand gestures (the movement of hand postures such as a flat palm on a table) and speech commands. Within co-located collaboration, I focus on ***small groups*** (mostly two but up to four people) who are physically co-located in the same space around a wall or table display. In particular, I am interested in how small groups use speech and gestures to serve as both commands to the computer and as awareness for other collaborators. That is, collaborator's actions create ***consequential communication*** that is used for awareness and coordination [Greenberg and Gutwin, 2004].

1.1 Background

Consider everyday life. Co-located collaborators often work with artefacts placed atop physical tables, such as maps containing geospatial information. Their work is quite nuanced, where people use gestures and speech in subtle ways as they interact with artefacts on the table and as they communicate with one another.

With the advent of touch sensitive wall and table displays, researchers are now leveraging their knowledge of how people collaborate over physical walls and tables to create appropriate technological interactions to support cooperative work. My own research is focused on advancing our understanding of multimodal co-located interaction, its nuances, advantages and tradeoffs. Specifically, I am interested in how two to four physically co-located people interact over a digital surface using hand gestures and speech, where these actions serve a dual purpose of interpersonal communication and computer input.

Given that physical tables and walls are already well used, why bother with digital ones? That answer is that digital content is becoming an important part of people's collaborative workflow. A few anticipated benefits are listed below:

1. Collaborators can collectively explore, interact with, and manipulate vast amounts of data in real time.
2. Updates in data can be processed and visualized in an automated way.
3. People can save and distribute changes and annotations on manipulated data.
4. People can easily interface with the digital media used in their current workflows.

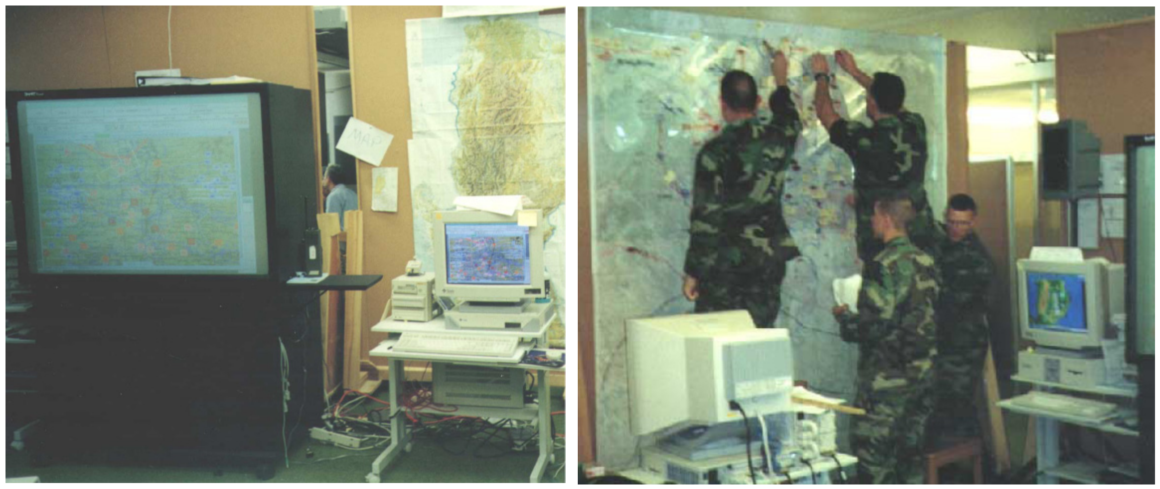


Figure 1.1. Digital displays in practice: state of the art military command and control systems in action (left) what commanders prefer (right) from McGee *et al.*, 2001

Despite these advantages large display interaction has still not seen widespread adoption in highly collaborative situations. For example, in military command and control environments, commanders are encouraged to use digital wall displays for planning and coordination due to the benefits of real time updates and rapid access to various layers of geospatial information (Figure 1.1, left). However, in practice these high tech systems are abandoned for a paper based map (Figure 1.1, right) that support multiple people manually placing and moving sticky notes along the map to visualize dynamic information [McGee *et al.*, 2001]. Within this context one of the many problems is that current operating systems are designed for a single individual using a keyboard and a mouse, even when these systems are placed on a large touch sensitive wall display the underlying application is incapable of supporting multiple streams of input. Even if multiple mice and keyboards are connected to a computer their inputs are merged into a single stream. Only this single stream of input is the easily accessible to application developers as multiple mice and keyboard require low level device programming. Thus, most applications are designed assuming there will be only one person interacting at any given time.

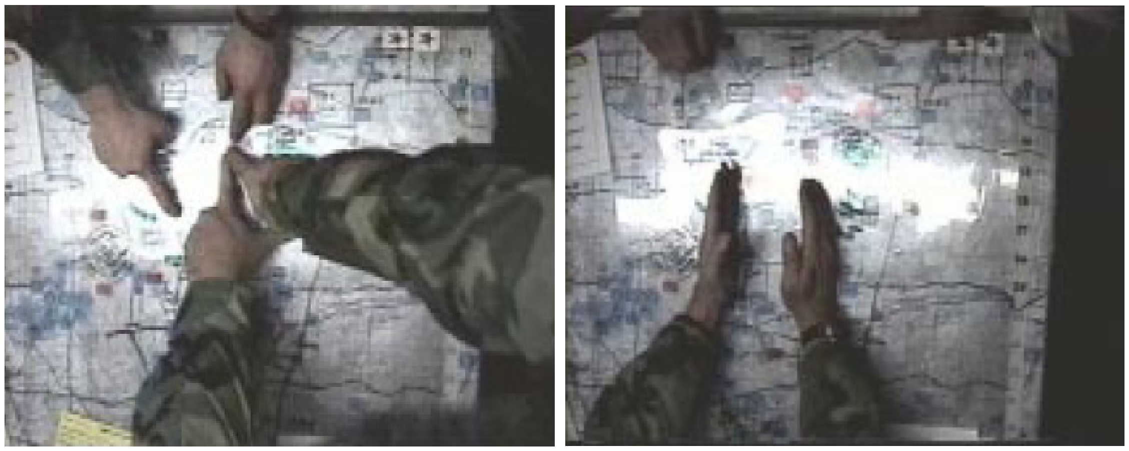


Figure 1.2. People interacting over paper maps: using multiple fingers to mark points of interest (left) using two hand sides to mark an area of interest (right) from Cohen *et al.*, 2002

The limitations of a single keyboard and mouse are illustrated when we observe how people collaborate on a physical table. For example, ethnographic studies [Cohen *et al.*, 2002] of tabletop collaboration in a military command and control setting reveals that people use multiple fingers to mark points of interest (Figure 1.2, left) and two hand sides to mark areas of interest on a map (right). This study also revealed 63-74% of all speech utterances were coupled with complementary gestures [Cohen *et al.*, 2002]. Other studies of different scenarios have revealed similar results [Tang, 1991]. The message is that in collaborative work people interact simultaneously using a combination of speech and expressive hand postures.

This dissertation explores the design and technical development of technologies that support multiple co-located people collaborating with multimodal speech and gesture commands over digital table. Multimodal interaction allows collaborators to leverage the expressive hand postures they use in practice on a digital system. These gestures could serve as both commands to the computer and as consequential communication to other collaborators. The vision is that the multimodal interactions could serve as a mechanism to interact with digital artefacts while still supplying the necessary consequential communication.

However, to examine the speech and gesture actions of multiple collaborators we need to overcome some of the fundamental limitations of using computers in a co-located setting:

1. Traditional small and low resolution desktop displays do not provide sufficient physical space for highly collaborative and problem solving situations.
2. Even if a large high resolution display is available, one person's standard window/icon/mouse interaction – optimized for small screens and individual performance – becomes awkward and hard to see and comprehend by others involved in the collaboration.
3. Current operating systems and applications are designed with single user assumptions. That is, only a single user is supporting, gesture input is restricted to a single point through a cursor, input from multiple mice and keyboards are merged into a single stream. These assumptions limit collaborators who are accustomed to using multiple fingers and two-handed gestures often in concert with speech.
4. It is difficult to design collaboration-aware multi user multimodal applications from the ground up. Existing applications contain functionality and robustness that is costly and time consuming to achieve in a research prototype.

The last point emphasizes the need to work with existing off the shelf applications. While it is difficult to develop multimodal co-located systems from the ground up, initial explorations can be facilitated by leveraging existing software. For this reason, much of the earlier chapters of this thesis focus on wrapping existing single user applications to work on a multi user multimodal digital table.

To introduce the content of my thesis I first describe the context of this research as it fits within the domain of human-computer interaction. Then, I detail the research objectives of this dissertation and conclude with an outline of the chapters in this thesis.

1.2 Research Context

This dissertation explores the design and technical development of technologies that support multiple co-located people collaborating with multimodal speech and gesture commands over digital table. Figure 1.3 illustrates how this research fits into the broad context in human-computer interaction (HCI). Within HCI, my research pertains to computer-supported cooperative work (CSCW), technologies designed to support groups of people working together. The next refinement narrows my focus to technologies that support co-located collaborative work, where collaborators are working in the same place at the same time. My focus can further be streamlined to include only those co-located collaborative environments that use multimodal gesture and speech input. My dissertation is motivated by and builds upon the lessons learned from theories, empirical and ethnographic research of group work and bridges a range of perspectives: human computer interaction, human factors, social factors/psychology, cognitive psychology and technological applications.

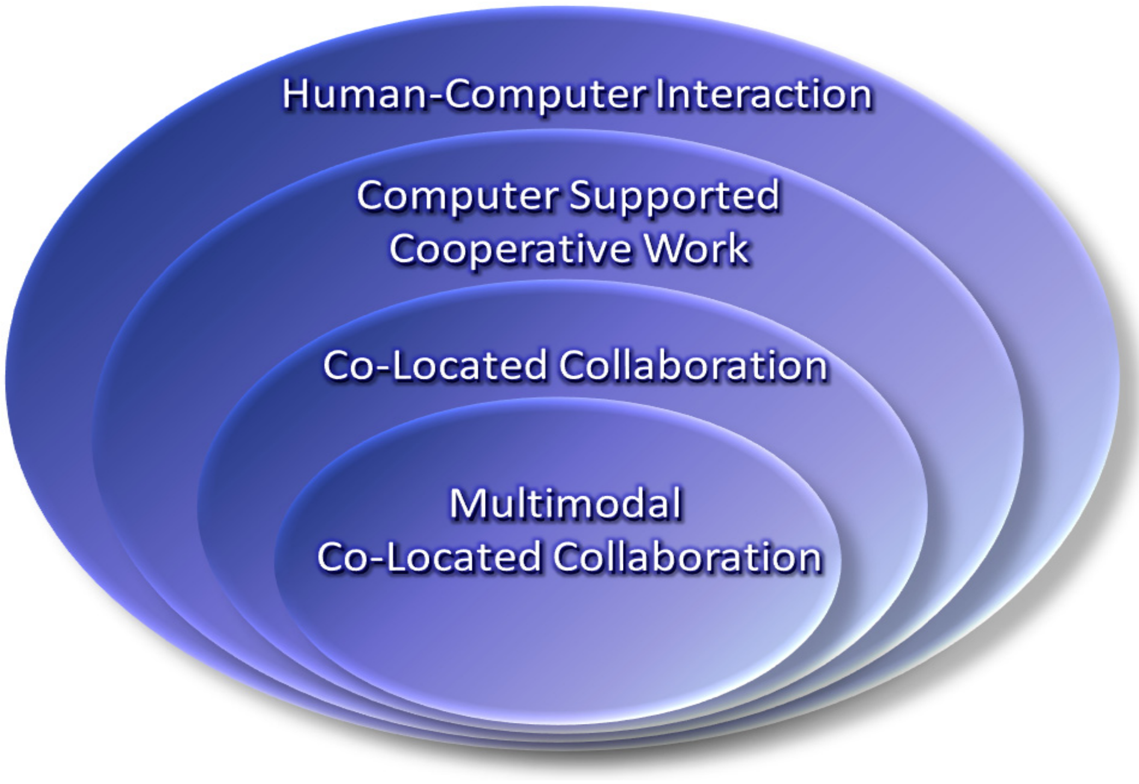

Human-Computer Interaction
Computer Supported Cooperative Work
Co-Located Collaboration
Multimodal Co-Located Collaboration

Figure 1.3. The Context of my Research

1.3 Research Problems

Multimodal Co-located interaction is a new area that is largely unexplored by the prior research. This thesis explores a breadth of different problems related to multimodal co-located interaction. In this section I describe seven research problems in the area of multimodal co-located interaction. Some of these questions address basic questions (e.g.

what benefits does multimodal co-located interaction provide? How do we implement a multimodal co-located system?), others address observations of people using multimodal co-located interaction (e.g. how do pairs collaborate with multimodal commands over a digital table using existing applications? What is the performance cost for using multimodal commands for selection over a large digital wall?), and still others explore the interaction possibilities of multimodal co-located interaction. In this thesis I will explore how we support co-located multimodal parallel work atop two types of groupware genres: collaboration-transparent and collaboration-aware systems [Lauwers and Lantz, 1990]. Collaboration-transparent groupware leverages the functionality of existing commercial single user software to let multiple people work with it; the underlying software is unaware of this collaboration. In contrast, collaboration-aware groupware understands that multiple participants are using it and directly supports collaboration.

The goal of this dissertation is to provide initial insights and explorations in multimodal co-located interaction to direct future research. At the end of this thesis (Chapter 9) I will summarize what has been learned about multimodal co-located interaction and I will describe how future research could leverage the lessons learned in this dissertation. The seven research problems addressed in this thesis are described below.

Problem One. We have not established why it would be useful to use speech and gesture commands in multi user co-located collaboration.

The first step in exploring multimodal co-located interaction is to motivate the use of speech and gestures interaction in a co-located setting. Specifically, we need to synthesize our understanding of speech and gesture use in a practical setting. We also need to understand the benefits of multimodal interaction in a single user setting and examine how this affects collaborators in a group work setting.

Problem Two. We have not designed multimodal interfaces for co-located interaction over collaboration-transparent single user applications.

Understanding the benefits if Problem 1 would assist the design of speech and gesture interfaces to leverage the benefits of multimodal interaction in a co-located setting. As mentioned earlier, it is difficult to build a collaboration-aware multi user multimodal application from the ground up. Thus by wrapping existing single user applications using a multi user multimodal speech and gesture interface we can explore interface design issues over applications with substantial amounts of functionality.

Problem Three. We have not explored how people use speech and gesture commands wrappers over existing single user applications on a digital table display.

After we have developed collaboration-transparent multimodal wrappers atop of existing single user applications we need to observe how people actually use speech and gesture commands for collaborative work. One of the goals is to determine if people use speech and gesture commands in much the same way as they do over physical tabletops as commands to the computer differ from the casual conversation of collaborators.

If multimodal speech and gesture commands can serve as both commands to the computer and as awareness to other collaborators designers can leverage this fact in future multimodal co-located systems. We also have not explored the potential interleaving of speech and gesture commands across collaborators in the co-located setting.

Problem Four. We have not implemented multimodal co-located systems that recognize the speech and gesture commands of multiple people.

In order to explore multimodal co-located interaction we must resolve the technical issue of implementing a multimodal co-located system over a digital table. In particular we need tools to recognize the speech and gesture commands of multiple people simultaneously before we can begin to explore multimodal co-located interaction. We do not know how the speech and gesture commands of multiple people should be mapped to keyboard and mouse actions over existing collaboration-transparent applications. Also we do not know how to create tools to support rapidly prototyping of collaboration-aware multimodal co-located systems.

Problem Five. We have not examined the performance cost of using speech and gesture commands for selection over a large digital wall display.

If co-located collaborators are to leverage multimodal speech and gesture commands for their collaborative work, it would be beneficial to understand the performance cost of using these multimodal commands in a co-located setting. On a large wall display, a dense selection space such as a cluttered desktop can be made sparser by using speech for selection. However, we have not examined if the cost of thinking about the selection filtering and the act of saying the selection filter outweighs the benefit of using speech for to filter the selection space. In particular, we have not evaluated how speech filtering will affect selection speed, accuracy, and preference when compared with gesture-only selection techniques.

Problem Six. We have not explored how parallel work can be supported over existing applications using a collaboration-transparent multimodal digital table.

Existing applications provide a breath and depth of functionality that would be too costly and time consuming to be practical in research prototype. However, these applications are limited by the single user assumptions built into current operating systems. If multiple people try to work in parallel the single user system will not know how to handle their inputs correctly. Simultaneous action is a requirement of parallel work. We do not know how to support parallel work over existing applications on a collaboration-transparent digital table display.

Problem Seven. We have not explored how multimodal co-located interaction can be applied to a collaboration-aware multimodal co-located groupware system. What design issues arise in the collaboration-aware setting?

Since we have yet to implement a collaboration-aware multimodal co-located application we have not explored the potential design issues of such groupware. The goal of this problem is to begin to understand some of the design issues with building collaboration-aware multi user multimodal systems that understand the inputs of multiple people and can leverage simultaneous input over a shared co-located surface. This will inform the design of future collaboration-aware multimodal co-located applications.

The previous seven research problems represent a breadth of exploration in the area of multimodal co-located interaction. The following section describes how I will approach each of these problems through a description of my research objectives.

1.4 Research Objectives

As multimodal co-located interaction is a new area in human computer interaction, many of my research objectives are best considered initial explorations in the area. The overarching goal of these research objectives is to inform the design and development of technologies to support multimodal co-located collaboration where multiple people interact with speech and gestures over a digital table.

Objective One. I will distill existing theories, empirical and ethnographic studies into a set of behavioural foundations that inform the design of multimodal co-located systems and outline individual and group benefits.

To motivate multimodal co-located interaction, one first needs to understand the natural behaviours of people engaged in collaborative work. I will perform a survey of existing theories of team work, empirical, and ethnographic studies to establish a set of behavioural foundations. I will break down the related literature and outline individual and group benefits for multimodal co-located interaction. This summary provides motivation for adding multimodal interaction to co-located environments. It also forms the basis of the design of the multimodal co-located systems developed in this thesis.

Objective Two. I will develop collaboration-transparent multimodal co-located wrappers over existing commercial applications.

Using the design implications and behavioural foundations developed in Objective One and the prototyping toolkit developed in Objective Four, I will develop several collaboration-transparent multimodal co-located interface wrappers atop of existing off the shelf applications for a digital table. By leveraging existing applications I will be able to rapidly prototype multimodal co-located applications that would otherwise be difficult and costly for one to develop from the ground up. I am specifically interested in mimicking the speech and gesture actions observed in ethnographic studies of collaborative work.

Objective Three. I will observe how pairs use collaboration-transparent speech and gesture wrappers over existing applications on a digital table.

As described in Problem Three, we do not know how people use collaboration-transparent speech and gesture wrappers over existing single user applications. I will observe how pairs use the speech and gesture wrappers developed in Objective Two for two collaborative tasks: trip planning and furniture layout. In particular I will investigate if multimodal speech and gesture commands serve only as commands to the computer or if they also serve as communication to other collaborators. These observations will have an impact on the design of future multimodal co-located systems.

Objective Four. I will develop a toolkit to enable rapid prototyping of multimodal co-located interactive systems.

As discussed earlier, existing operating systems and applications are designed with the assumption that there will be only one user per computer. This assumption introduces a number of technical hurdles that must be resolved to explore even the most basic multimodal co-located interactions. Using the experience gained from my Master's Thesis on creating a Single Display Groupware Toolkit [Tse, 2004c] I will develop rapid prototyping software to facilitate the multimodal co-located explorations of this thesis.

This objective consists of 3 sub-goals:

1. I will develop a gesture recognizer that recognizes different hand postures (e.g., arm, hand, five finger, fist, etc) and their respective dynamic movements (called gestures e.g., two fingers moving apart) for multiple people (two to four) on a digital table.
2. I will develop a multimodal integrator that fuses the speech and gesture commands from an individual or several people into system commands. I will leverage existing speech recognition technology to recognize voice commands and I will leverage some of the hand postures described in related literature (e.g., finger, palm, hand sides).
3. I will develop a utility to simplify the adaptation of commercial single user applications to a collaboration-transparent multimodal co-located environment, allowing one to rapidly prototype multimodal applications without the need to develop a working commercial system from the ground up. This will facilitate my initial explorations of multimodal co-located interaction.

This toolkit will be designed to support gestures and speech on existing commercially available digital tables. I also configure these tables in a custom manner to explore new interaction possibilities in Objective Six.

Objective Five. I will compare the performance of using speech filtering for selection on a large digital wall display to two commonly used gesture-only selection techniques.

As mentioned in Problem Six, we have not examined the performance cost of using speech commands coupled with gestures for basic tasks such as selection on a large digital wall display. I will compare speech filtered selection to two gesture-only selection techniques for large displays: ray casting (pointing to the wall with a single finger), and bubble ray (a technique adapted from Grossman and Balakrishnan [2005] that involves pointing to the wall with a single finger and having the closest item always selected). Specifically, I will measure selection speed, errors, and user preference between the three techniques. While this study will be performed for a single person, selection is a basic activity that will influence co-located activity as well.

Objective Six. I will design and implement a collaboration-transparent multimodal split view table to support parallel work over existing applications.

While existing applications are limited by the single user assumptions of current operating systems, parallel work can be supported over a shared digital table by projecting two separate views onto the digital display. A split view tabletop is a tabletop surface whose surface is split into two adjacent projected views. By connecting separate computers to each view multiple people can work simultaneously over each computer. Multimodal speech and gesture commands are used to augment awareness of collaborator's actions in the collaborative setting. I will describe different seating arrangements and software configurations for a multimodal split view tabletop. I will demonstrate multimodal split view tabletop interaction in practice through three case studies over existing applications.

Objective Seven. I will develop a collaboration-aware multimodal co-located system to explore issues for future designers.

Using the toolkit developed in Objective Four and the lessons learned from Objective Three, I will create a collaboration-aware multimodal co-located system from the ground up that explores concurrent multi user multimodal interactions. I will support parallel work by allowing simultaneous interaction over a digital table, where I will explore joint multimodal commands that leverage the simultaneous inputs of multiple people. The goal of this exercise is to reveal issues that future multimodal co-located system designers will likely encounter.

1.5 Organizational Overview

Chapter 1.	I introduce the concept of multimodal co-located interaction and the context of this dissertation. I establish my thesis objectives and provide an overview of future chapters.
Chapter 2.	I describe related technical innovations within the context of co-located collaborators. I also describe technologies in multimodal interaction specifically within the context of public actions that are perceptible to other collaborators.
Chapter 3.	I distill existing theories, empirical and ethnographic studies into a set of behavioural foundations to motivate multimodal co-located interaction. These behavioural guidelines provide the basis for the design of several collaboration-transparent multi-user multimodal wrappers around existing single user applications. A discussion of issues arising and lessons learned concludes this chapter.
Chapter 4.	I describe an observational study on how pairs of people used the collaboration-transparent speech and gesture wrappers described in Chapter 3. This revealed that people overwhelmingly used speech and gesture commands both as commands to the computer and as communication to collaborators. I explore how people interleaved speech and gesture commands, working in parallel despite the single user limitation of the underlying application.
Chapter 5.	I discuss the development of GSI DEMO, a tool for rapid prototyping of multimodal co-located interaction on a digital table. GSI DEMO allows existing off the shelf applications designed for a keyboard and a mouse to be used with speech and gestures on a digital table. It is the enabling infrastructure that allowed existing single user applications to be used with speech and gesture commands in Chapters 3 and 4. It also provides the

infrastructure needed to build multimodal co-located applications from the ground up and serves as a technical foundation for future chapters.

Chapter 6. I introduce an interaction technique for distant freehand pointing on a large wall sized display called the Speech-Filtered Bubble Ray. This technique allows one to point to a general area on the screen and use speech to filter the selection space. For example, someone could point to a large desktop and say "that PowerPoint file." A Fitts Law study comparing this technique to two gesture only selection standards found that the Speech-Filtered Bubble Ray technique was fastest, least error prone, and most preferred by participants.

Chapter 7. Since existing single user applications are designed to accept only a single stream of keyboard and mouse input, they are unable to support parallel work in co-located setting. I circumvent this limitation with the introduction of a multimodal split-view tabletop, a tabletop surface that is split into two adjacent projected views connected to separate computers. By using separate computers pairs can work in parallel over separate desktops over a shared tabletop surface. Multimodal commands provide enhanced activity awareness especially when people are working in parallel. I describe several physical and software configurations for split view tabletops and illustrate its use through three case studies.

Chapter 8. While previous chapters focused on leveraging existing single user applications, they were fundamentally limited by the assumption that a keyboard and mouse design would be used for interaction. In this chapter, I resolve these difficulties by creating a collaboration-aware multimodal co-located system on a digital table for industrial designers. Through the development of this system I encountered a number of design issues specific to multimodal co-located interaction. I describe these issues for the benefit of future multimodal co-located system designers. I also describe the infrastructure needed to support multi user multimodal interaction over multiple displays and input devices.

Chapter 9. I conclude the thesis by revisiting the research objectives described in Chapter 1 and describe how each objective has been accomplished within my thesis. Finally, I summarize the contributions of my thesis and describe several venues for potential future work.

Chapter 2. Technical Related Work

In this chapter I provide background of the technical explorations leading up to and motivating the work described in this thesis. This technical background is divided into two sections: co-location technology and multimodal technology. Co-located technologies are used to support collaborators in a face-to-face setting (e.g. a shared desktop). My review focuses on co-located interaction technologies where multiple people share a single display; this is called Single Display Groupware (SDG). I begin with an exploration of SDG desktop extensions. I then discuss how multiple people can interact over a large wall and tabletop display. Finally I examine interactive surface environments that may involve multiple wall and table displays used concurrently by multiple collaborators.

The second part of my review examines multimodal speech and gesture interaction within the context of the co-located interactive surfaces discussed earlier. To focus this I limit my discussions to explicit multimodal interaction (as opposed to implicit, ambient, and perceptual multimodal interfaces), where people use speech and gesture input to directly manipulate system actions and objects. I begin with multimodal desktop extensions, and continue to multimodal interaction over large wall and table displays. Finally, I discuss multimodal interaction within the context of augmented and virtual reality environments.

The goals of this chapter are threefold. First, I aim to familiarize the reader with previous technical explorations in co-located and multimodal interaction. Second, I provide the reader with context valuable for understanding the research contributions of later chapters. Finally, I emphasize that this thesis is about the combination of multimodal and co-located interaction rather than advancing research of a single domain.

2.1 Co-Located Interaction Technology

This section describes input approaches to supporting face to face collaboration. I focus on systems where people share a single display, called Single Display Groupware (SDG). While people can easily share a single input device such as a mouse through turn taking, I pay particular attention to the input methods that allow ***multiple*** people to interact ***simultaneously*** in the co-located setting as this is a requirement for supporting parallel work. I begin with a discussion of input extensions to traditional desktop computers. Next I describe the evolution of interactive wall and table displays for co-located interaction. Finally I describe interactive surface environments that leverage multiple wall and table displays.

2.1.1 SDG Desktop Extensions

When people work side by side on a desktop computer they can collaborate over existing applications by sharing a single keyboard/mouse and taking turns handing off the input devices to each other. This can be cumbersome if the collaborative task involves many input device exchanges. An alternative is to connect multiple mice and keyboards into a single machine. While this removes the physical constraint of sharing a single mouse, a technical constraint is introduced by current operating systems. That is, current operating systems assume that there will be only one user per computer, thus the input of multiple mice are merged into a single stream. If two people try to move their mice simultaneously the cursor seems to move erratically around the display. Many early gaming systems resolved this conflict by designing custom built systems from the ground up that could leverage multiple input devices (game controllers, joysticks, personal digital assistants). Research in SDG attempted to resolve such conflicts over personal computers by detecting the input of multiple mice as separate streams and providing a cursor for mouse (as illustrated in Figure 2.1).

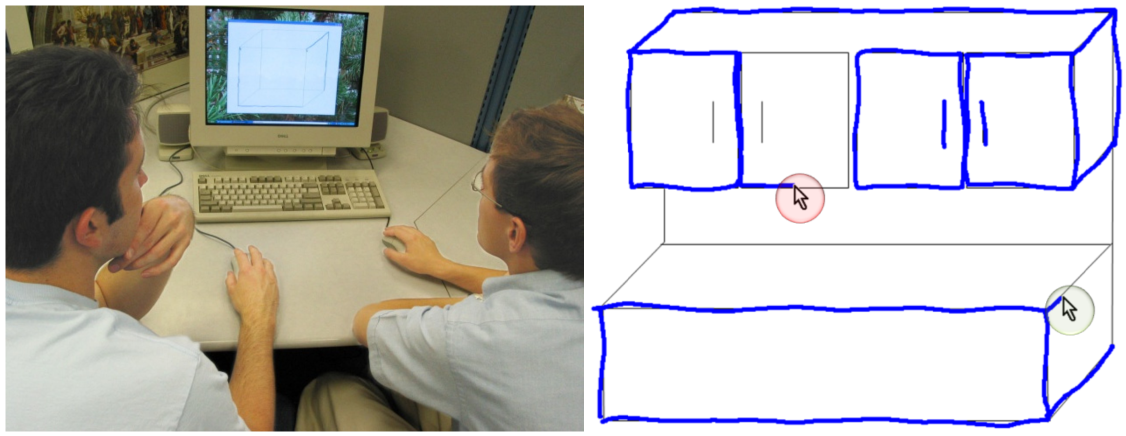


Figure 2.1 Multiple people working over a single display using multiple mice (left), a multiple mouse drawing application (right) images from Tse *et al.*, 2004a.

SDG with multiple mice and keyboards. One way of circumventing merged mouse input is to modify the operating system or add programming utilities so that the programming environment can support multiple simultaneous inputs. This was first done by Bier and Freeman's [1991] Multi device Multi user Multi editor (MMM) system. To support multiple mice they modified the operating system to read the input of two mice directly from a serial port. This input was then sent to a modified window manager that showed individual cursors (one per mouse) and modes (e.g., moving mode, drawing mode) using a status window located on the bottom left corner of the screen. MMM set the standard for future SDG research systems.

At the toolkit level software was developed to enable rapid prototyping of SDG applications that provided input from multiple input devices as separate streams in high level programming langauges. For example, MID [Bederson and Hourcade, 1999] and SDG Toolkit [Tse *et al.*, 2004b] provided programmers with a simple application programmers' interface to access input from multiple mice. This is detailed in the SDG Toolkits section.

At the interface level for example, Bricker *et al.* [1999] expanded on MMM by exploring cooperative *joint* actions. This uses the input of multiple people to control a single artefact, for example, simultaneously manipulating control points on a curve. Latulipe *et al.* [2006] and Tse *et al.* [2004b] explored bimanual interactions involving a single person using two mice. KidPad by Bederson *et al.* [1996] improved upon the status windows of MMM by visualizing states through local tools. As illustrated in Figure 2.2 (right), local tools are icons (such as a red crayon) that can be picked up by clicking on the icon. When local tools are picked up they change the current mode (to drawing) and change the image of the mouse cursor to the local tool. When released the current mode is reverted (to moving) and the local tool icon is left at the position of the mouse cursor. Tse *et al.* [2004b] extended the MMM concept by enabling multiple keyboards as well as mice to be used simultaneously, to provide high performance cursors to end programmers, to supply an infrastructure for building multi-user widgets.

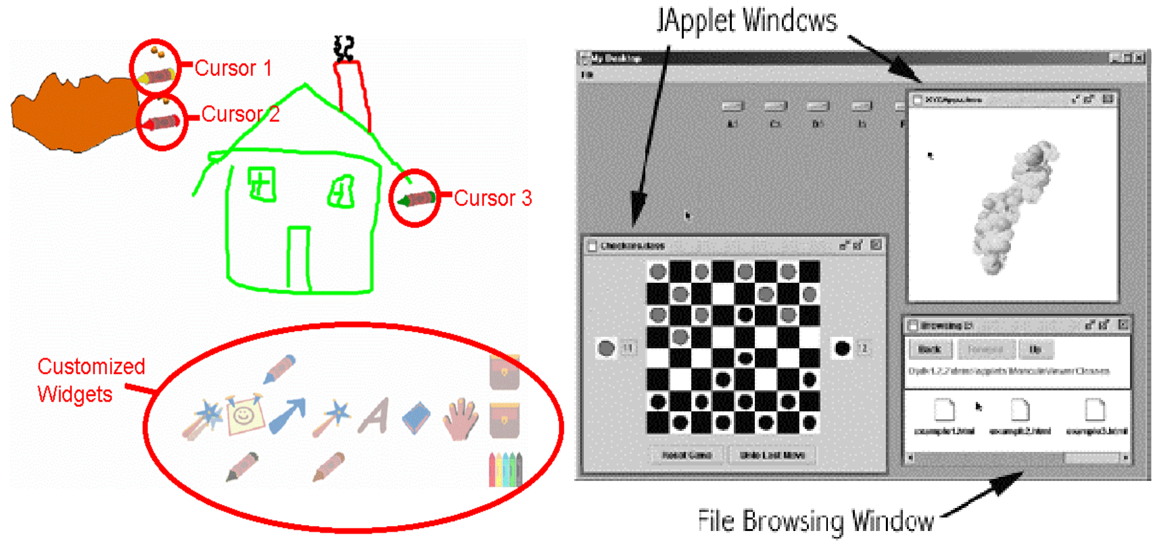


Figure 2.2 SDG Applications: KidPad from Bederson *et al.*, 1996 (left), Interacting with multiple windows using MID Desktop, from Shoemaker and Inkpen, 2000 (right).

SDG over existing applications. It is difficult and costly to build truly useful applications from the ground up. Consequently most research prototypes have insufficient functionality and robustness to be used in a practical setting. For this reason, there has been significant interest in leveraging existing off the shelf applications in an SDG setting. These are applications intended for a single user, and only recognize a single mouse and/or keyboard as input. The idea is to somehow wrap these applications so that people can use multiple input devices where the wrapper takes that input and transforms it into a single mouse/keyboard stream understood by the application. As we will see in future chapters, we use a similar approach with multimodal input.

MID Desktop is a desktop navigation system that allowed any Java applet to be run as an SDG application with individual cursors [Shoemaker and Inkpen, 2000]. The Java applets running in the MID Desktop environment are sent a single stream of input serialized by MID Desktop. Multiple people could work over separate Java applets on the same computer simultaneously without conflicts. As illustrated in Figure 2.2 one person could move pieces on a game of checkers while another could search for documents using a file browser. However, conflicts would arise if multiple people tried to interact with a single applet simultaneously. For example, if two people moved the same scroll bar in an applet simultaneously the scroll bar position would jump erratically.

A somewhat similar approach was done by Sing, Gupta and Toyama of Microsoft Research India. They enabled multiple people to interact over two independent computer desktops on a single computer by splitting the digital display vertically (http://research.microsoft.com/research/tem/). This is akin to having two side by side computers as a way to reduce costs for technology deployment in developing nations as multiple people could work in parallel using their own keyboard and mouse on half of the screen.

The Glove Programmable Input Emulator (GlovePIE) developed by Carl Kenner (http://carl.kenner.googlepages.com/glovepie) used a specialized mouse driver to capture the input of multiple mice and provided a scripting language so that a programmer could customize how the input from multiple people would be converted into a single stream. For example, one could ignore the input of all but the primary mouse or they could explicitly release control of the mouse using a mouse button (some techniques discussed in Tse *et al.*, 2004b).

SDG Toolkits. As mentioned earlier, current operating systems by default merge the input of multiple mice into a single stream. Only this merged input stream is easily accessible to end programmers. Thus if a programmer wants to handle multiple input devices in a custom application they must either build their own device drivers or access some low level application programmer's interface (API) and do considerable extra work adding on capabilities provided for free with a standard system mouse (e.g. drawing the cursor, transforming mouse movements to window coordinates).

To enable rapid prototyping of SDG applications Bederson and Hourcade [1999] developed the Multiple Input Devices (MID) toolkit. MID toolkit simplified access to multiple mice by internally managing low level APIs and presenting a simple interface in Java.

SDG mouse events were presented to end programmers in a form identical to regular Java mouse events with the addition of an identifier specifying the mouse generating each event. Of course MID had limitations. Multiple cursors had to be drawn manually and the low level APIs did not function past Windows 98. Yet despite these limitations, very compelling applications such as KidPad (described earlier) were developed.

SDG Toolkit extended the functionality of MID toolkit to significantly simplify the programmer's task. It automatically provided multiple high performance cursors (one for each mouse), mouse input relative to the application window a person was using, support for mice used on a tabletop display, support for multiple keyboards, and the ability to create SDG widgets (e.g., buttons, sliders) that understood simultaneous interaction from multiple people [Tse *et al.*, 2004].

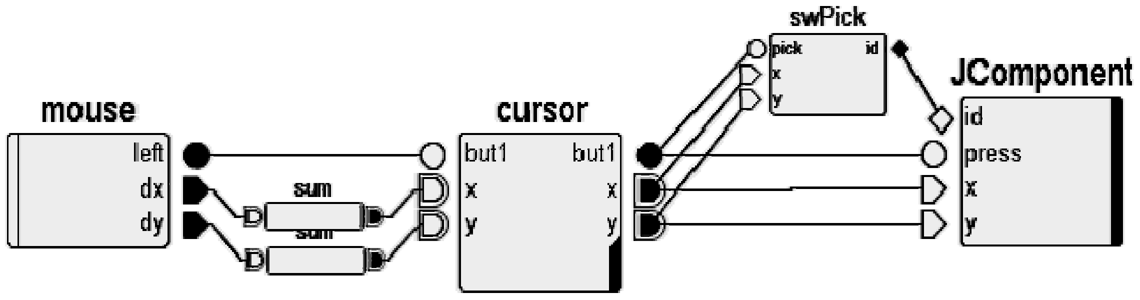


Figure 2.3. Using visual programming to map application functionality in the ICON toolkit, from Dragicevic and Fekete, 2004

Several other toolkits provided support for many novel input devices in an SDG setting. The Beach architecture [Tandler, 2003] provided a networking approach for multiple wall and table displays as well as individual tablet computers (described in §2.1.4). As illustrated in Figure 2.3, the Input Configurator (ICON) toolkit [Dragicevic and Fekete, 2004] allowed end users to map input from multiple devices to program functionality using visual programming. Programmers could draw links between mouse input, cursor visualizations and program input. When executed the input would follow the flow diagram specified through visual programming. More recently, many other toolkits were developed to support specific capabilities for tabletop displays. For example, Diamond Spin [Shen *et al.*, 2004] provided artefact rotation and movement on a tabletop display. The Buffer Framework focused on the graphical issues of supporting many objects rotated at arbitrary angles on a large high resolution digital table [Isenberg *et al.*, 2006]. Digital wall and table interactions are discussed further in subsequent sections.

2.1.2 Digital Wall Interaction

Large upright displays are often used for presentation, discussion and collaboration as they afford viewing by multiple people. Yet, in order for multiple people to collaborate over a large display, suitable input devices are needed. Arcade controllers, light guns, and racing wheels have long been used to manipulate large displays in competitive games, yet outside of research systems, they are seldom used for collaborative work. In this section, I focus on expressive interactions designed to support collaborative work.

There are three primary approaches to let people manipulate work artefacts on an upright display: people can walk up to the display and directly touch it, they can interact with the display at a distance using distant freehand pointing, or they can use indirect devices such as mice and PDAs to manipulate items on a large display. My discussion focuses on direct touch and distant freehand pointing as these actions are the most publicly visible to other collaborators.

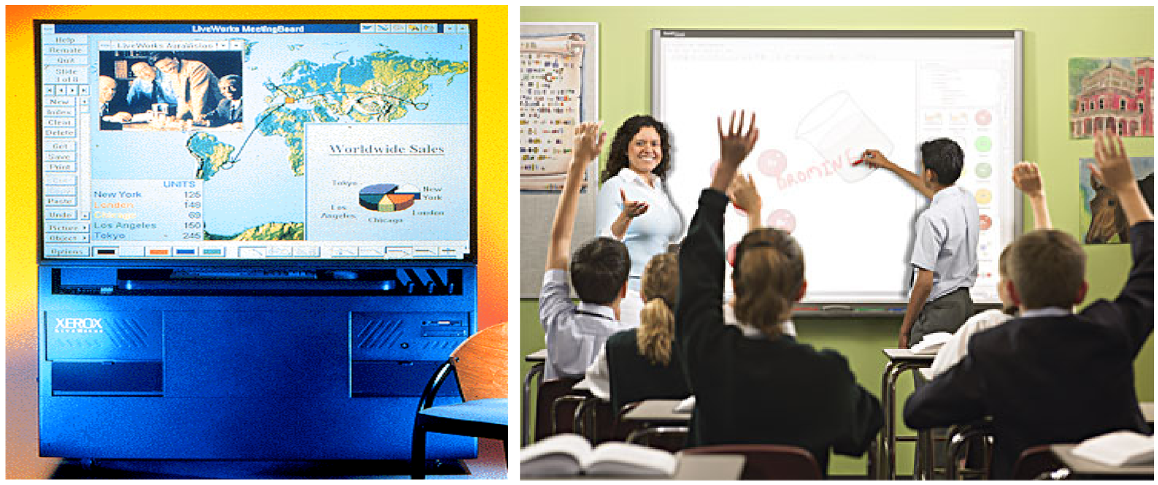


Figure 2.4 Single point interactive wall displays: Xerox Liveboard, Image courtesy of Palo Alto Research Center, photographed by Brian Tramontana (left), single touch analog resistive Smart Boards from www.smarttech.com (right)

Direct Touch Interaction. Direct manipulation on the surface of a wall display is typically done with a system pen or bare fingers. As illustrated in Figure 2.4 (left) LiveBoard was an early pen operated digital white board that supported a single point of contact [Elrod *et al.*, 1992]. Commercial single touch displays are now available. Mimio provides pen tracking technology using acoustic ultrasonic sensors (www.mimio.com) which can be attached to any flat surface. As illustrated in Figure 2.4 (right) Smart Technologies (www.smarttech.com) provides a number of analog resistive touch surfaces for projected displays that can detect a point or a finger. With the exception of later versions of the Smart DViT board, each of these systems only support a single touch. Thus they do not support multiple users except through turn taking. Some researchers have circumvented this issue by tiling several large single touch displays. If only one person uses each display tile, people can then work simultaneously on their respective portion of the tiled workspace. This approach was used in the Dynawall project [Streitz *et al.*, 1999] where several Smart Boards were tiled to create a large wall surface (Figure 2.5, left). As seen in the Figure, each person must avoid working on the same Smart Board, thus collaborators have to stay physically distant from one another. While reasonable for loosely coupled individual work, this separation would likely discourage tightly coupled joint work.

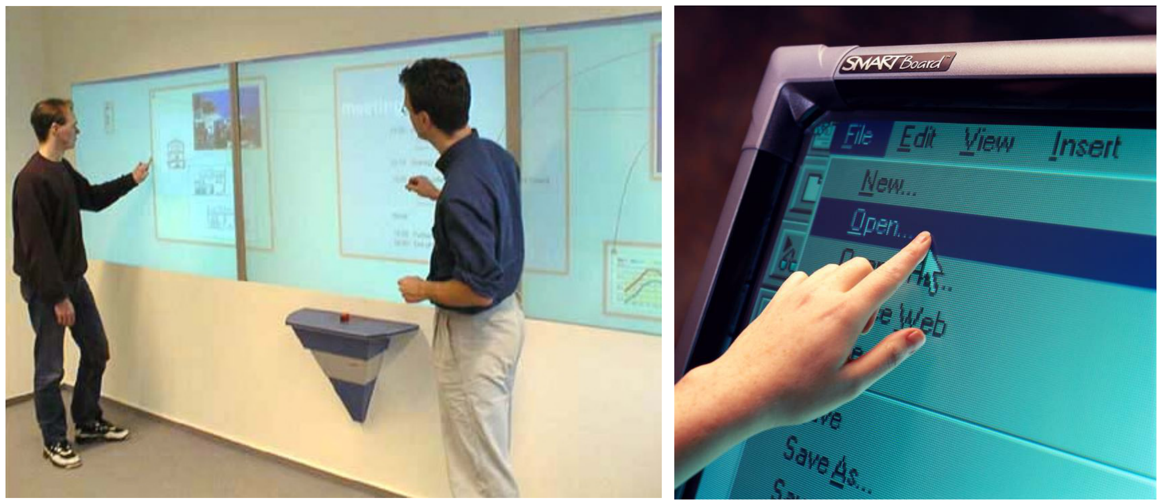


Figure 2.5. Multi-touch wall displays: Multiple tiled Smart Boards in the Dynawall from Streitz *et al.*, 1999 (left) dual touch DViT (right), from www.smarttech.com

Recent advances in touch technology have enabled wall displays to detect multiple simultaneous touches. All technologies differ substantially, each having different strengths and weaknesses. Our first example is the Digital Vision Technology (DViT) from Smart Technologies, which supports up to two simultaneous touches (Figure 2.5, right). It uses an infrared camera on each corner of the board that observes an array infrared LEDs on each side of the display. When a pen or finger is moved over the board, images are analyzed by the four infrared cameras and the points of contact of calculated. The benefit is that DViT can be overlaid atop of any display technology (including flat panel televisions) making it quite versatile. However, the side camera approach can result in occlusion when many fingers are used. This is why the current DViT API is limited to reporting two points of contact.

A second example is the Frustrated Total Internal Reflectance (FTIR) techniques such as those by Han [2005] (Figure 2.12, right), which allows an arbitrary number of simultaneous touches to be detected. FTIR works under the principle that light projected along the sides of large sheet of acrylic will bounce internally until one contacts the surface (by touch or pen) and frustrates the light so that it projects perpendicular to the surface. This frustrated light can be detected by infrared cameras behind the surface with very little external calculation required. The benefit is that an arbitrary number of points can be detected. The limitation is that the cameras must be able to see the entire touch surface, making it difficult to use in a flat panel setting.

Distant Pointing Interaction. An alternative to direct touch interaction would be to point at the large display using a hand or laser pointer. Distant pointing can be beneficial when it is not convenient to walk up and touch the display (e.g. when seated in a meeting room).

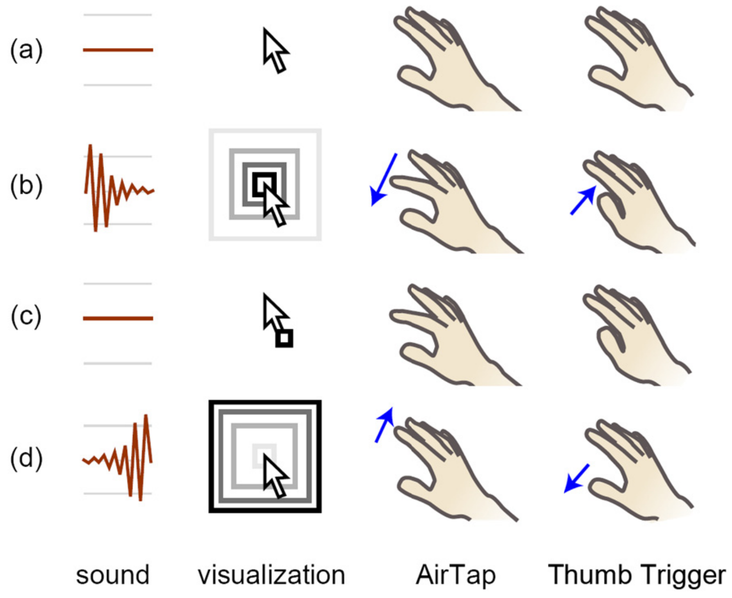


Figure 2.6 The AirTap and ThumbTrigger interaction techniques, from Vogel and Balakrishnan, 2005

A simple approach is to use laser pointers as input, Olsen and Nielsen [2001] developed Xweb, a system that tracked the position of a single laser pointer using multiple web cameras. This work was extended by Vogt *et al.* [2003] to support multiple laser pointers, each distinguished using distinct blinking patterns. One issue is how selections versus pointing actions are indicated. In laser pointer systems (e.g. Vogt *et al.,* 2003), selection is typically performed by pressing a button or flipping a switch.

Explicit motions are required to trigger a selection when using whole hand pointing. Vogel and Balakrishnan introduced two techniques for selecting with a hand: with the ***thumb trigger*** the user presses their thumb against their hand to perform a selection (Figure 2.6b). With the *air tap*, the user moves their index finger to indicate selection (Figure 2.6d). A visual and auditory visualization was also provided to indicate the cursor location on the large vertical display.

A variety of other techniques have been used to track the position and direction of a hand in free space. Some techniques include using data gloves (e.g., 5DT data gloves www.5dt.com), magnetic tracking (e.g., Kaiser *et al.*, 2003), visual marker tracking (e.g. Vogel and Balakrishnan, 2005), infrared imaging (e.g., Wilson, 2004), and strict computer vision (e.g., Wren *et al.*, 1997).

Indirect Interaction. As opposed to public direct touch interaction and distant freehand pointing actions, private indirect interactions produce minimal amounts of awareness information to other collaborators, this is called consequential communication. As mentioned earlier, game controllers used with console games over large television screens typically do not demand consequential communication, as players are supposed to be seated facing the television rather than each other. Researchers have also used individual personal digital assistants to manipulate artefacts on a large vertical wall. Myers *et al.* [1998] developed the Amulet system to allow multiple collaborators in a meeting room use a drawing application on a large shared screen.

Finally, researchers have also used a camera to track the positions of a hand on a physical table with a solid background. Using blue screening techniques one can easily remove the solid background to reveal an image of the hand for use with a hand tracking system. Roussel [2001] allowed the hand to be overlayed on top of a tradition desktop computer in the VideoSpace system. The user could use their index finger as a replacement for a traditional mouse cursor to interact with a desktop computer. Malik *et al.* [2005] extended this concept for manipulating items on a large wall display using a set of hand postures (fist, one finger, two finger pinch, etc). Similar to VideoSpace, this system would also visualize a person's hand on the digital wall display. In both systems, hand postures were performed on a physical table with a solid background while an overhead camera was used to monitor the individual's hand postures. Collaborators familiar with these hand postures could see them being using on the physical table or projected on the digital wall for awareness of what others were doing.

2.1.3 Digital Table Interaction

A digital wall turned horizontal becomes a tabletop display. Yet tables have considerably different affordances and uses than walls, thus technology for tables has focused on supporting a different albeit overlapping set of possible actions than those of wall displays. For example on tables, people often lean over them, they place objects on them, work across from one another, orient items towards themselves and others, and carefully observe the facial expressions of others [Scott *et al.*, 2003 and Tang, 1991]. Horizontal displays are valuable for collaborative work because they provide a common surface that allows people to monitor the digital display and view the body language of others simultaneously.

There are many types of digital table environments such as single user drafting systems (e.g. Wellner 1991, Buxton *et al.*, 2000), tables that use tangible interfaces (e.g. Ullmer and Ishii, 1997, Patten *et al.*, 2001), augmented and virtual reality tables (e.g., Krueger *et al.*, 1995, Agrawala *et al.*, 1997) and large screen, direct touch tabletop displays (e.g., Dietz and Leigh, 2001, Rekimoto, 2002, Han, 2005). This dissertation explores the design and technical development of technologies that support multiple co-located people collaborating with multimodal speech and gesture commands over digital table. For this reason my discussion focuses around direct manipulation using pens, tangible objects (such as wooden blocks), fingers, and hands. I begin by describing pen based digital table interaction, then I describe digital tabletop interaction with tangible objects, and finally I describe direct touch interaction on a digital table.

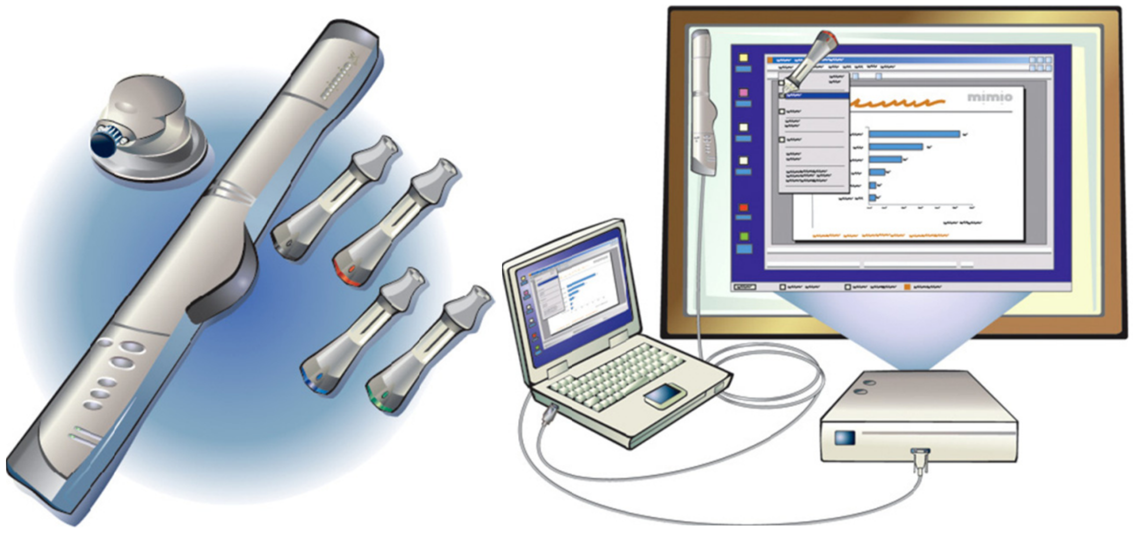


Figure 2.7. The Mimio pen tracking system: mimio pens and ultrasonic tracer (left) Mimio system connected in a front projection (right) from www.mimio.com

Front projection versus rear projection: As mentioned earlier a digital wall display turned horizontally becomes a tabletop display. When using a computer projector there are two method of projection: front projection and rear projection. Front projection works by top projecting a digital display onto a traditional table (Figure 2.7, right). This can be done by mounting a projector on a stand and pointing it towards a table. This is a simple way to assemble a tabletop display but it can result in shadows over the table surface when interacting from above (as illustrated in Figure 2.8, left). Conversely rear projection involves projecting an image from below to a clear piece of glass or plastic table surface, a diffuser such as rear projection paper ensures that the image appears on the tabletop surface. Rear projection has the advantage that shadows from the hands and arms are removed but it requires a large amount of space or specialized mirrors underneath the table surface.

Pen Based. As illustrated in Figure 2.7, the Mimio pen (www.mimio.com) is an ultrasonic tracker than can be used enable pen interaction with a tabletop display [Rogers and Lindley, 2004]. Mimio works by using a receiver mounted on the table (Figure 2.7, left) to listen to ultrasonic signals from a battery powered pen ultrasonic pen. When a person writes with the pen a switch is activated that sends a signal from the pen to the receiver. Tracking is limited to writing with a single pen directly on the table surface. Alternatively one can turn a SmartBoard display horizontally to interact with a pen or a finger.

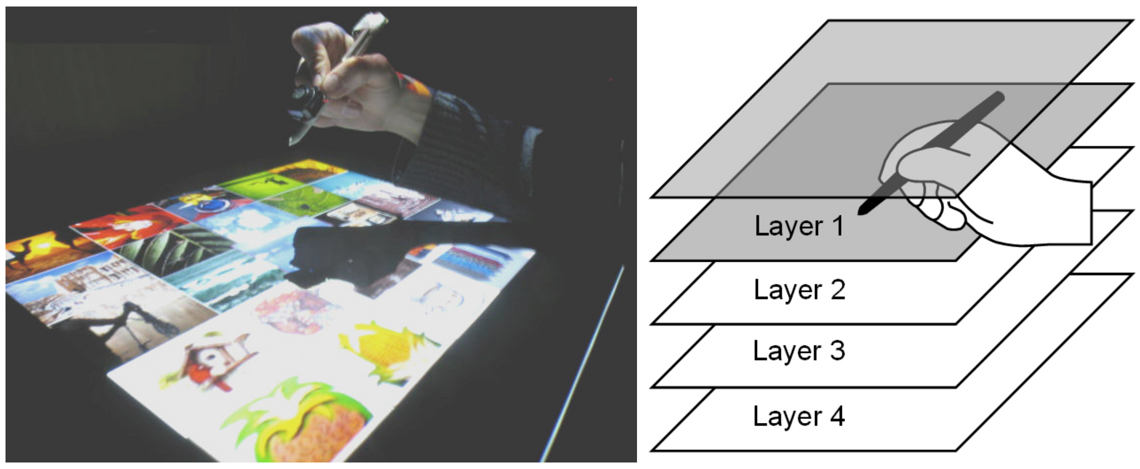


Figure 2.8. Using a pen to interact with layers above a digital table, from Subramanian *et al.*, 2006

Using 3D pen tracking, the position, orientation, and roll of the pen can be leveraged over a digital table. For example Subramanian *et al.* [2006] demonstrated interaction above the tabletop surface using a magnetically tracked pen. Gestures above the digital table could be recognized and used by the system. Figure 2.8 (right) illustrates multi-layer interaction, where each layer can be used to select a different modifier for a drawing application. For example, Layer 1 (16-12 centimetres above the table) represents the size of the pen stokes, Layer 2 (12-8cm) represents the colour, Layer 3 (8-4cm) represents the pen shape, and Layer 4 (4cm and below) is used to actually draw on the surface.

Digital pens (such as Logitech's IO pen, www.logitech.com) are regular pens that have an embedded camera looking down at what the user is writing. These cameras track position codes printed on large sheets of paper to allow regular ink strokes to be stored in a digital form. Researchers have begun to use digital pens for high resolution interactions with digital tables. By replacing the ink pen tip with a plastic tip, multiple people can draw onto large sheets of position coded paper without leaving any permanent marks. Instead, the ink strokes can be visualized by top projecting a digital image onto the paper using a projector. For example, Haller *et al.* [2005] demonstrated a drawing application that allowed multiple people to create annotations using wireless digital pens. Two overhead projectors pointing down to large sheets of position coded paper allowed pen strokes to be visualized on the table. They also showed how tools could be selected using small sheets of position coded paper placed onto a portable piece of plastic. By touching specially marked regions with the pen, a user could change their stroke thickness and colour in the drawing application.

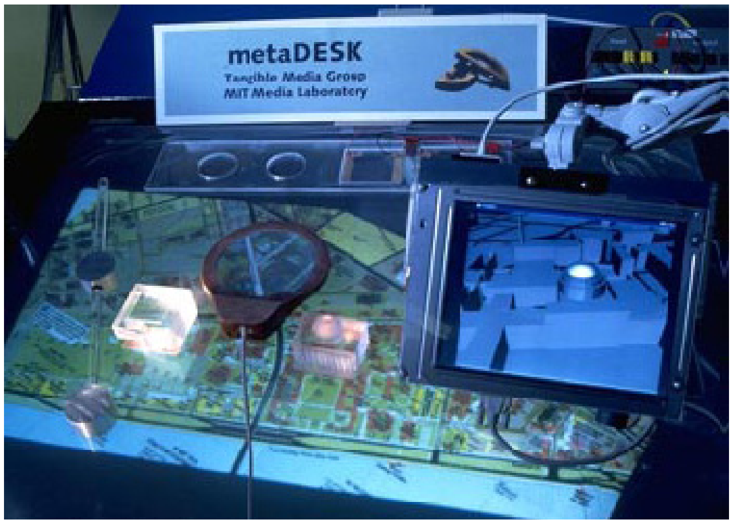


Figure 2.9. Physical objects placed atop the metaDESK, from Ishii and Ullmer, 1997

Tangible Objects. The act of moving tangible (or physical) objects on a digital table can produce consequential communication that others can use for awareness in a co-located setting [Gutwin and Greenberg, 2004]. MetaDESK by Ishii and Ullmer [1997] allowed physical objects to be tracked on a digital table. As illustrated in Figure 2.9, a variety of objects could be placed on the table including passive objects such as the ruler (left) and plastic buildings. Manipulating the physical object would result in a similar change to the table display. For example, moving two points on the ruler would result in a corresponding rotation, pan and zoom of the digital map. Thus the tangible manipulations could be used to modify artefacts such as a virtual map of a University campus.

Objects containing digital displays such as the magnifying glass and tablet computer could also be tracked (Figure 2.9, right). These digital displays could reveal additional information such as a 3D view of the buildings on campus or roads, restaurants and other geospatial content. Rekimoto *et al.* [2002] extended the concepts of metaDESK by allowing dynamic digital content to be shown underneath clear glass tiles. This gave the illusion that passive objects could display dynamic content. For example, by placing a clear glass tile on a tabletop a video could be projected underneath it giving the illusion that the glass tile holds video content.

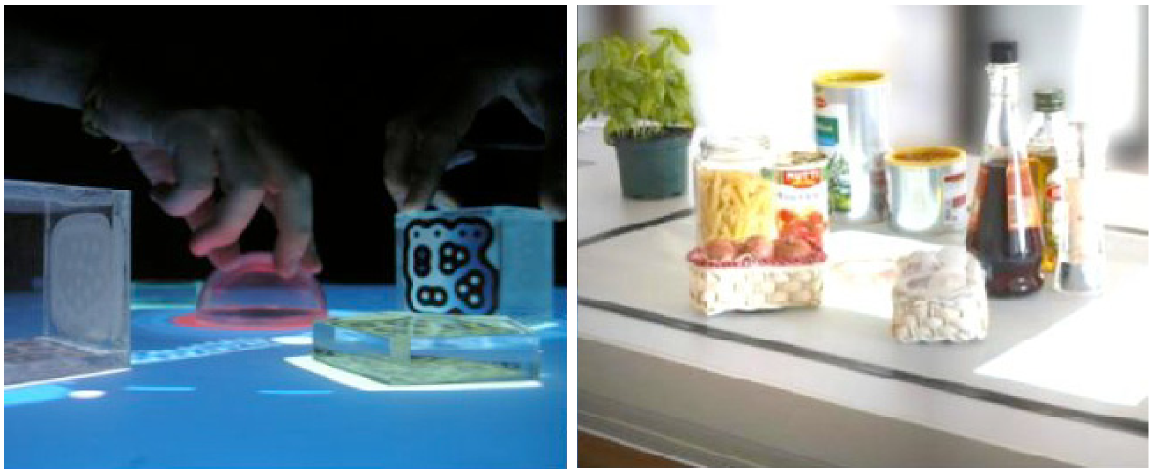


Figure 2.10 Tangible tabletop interaction using fiducial markers: the recTable (left), recipe-table (right), from Kaltenbrunner and Bencina, 2007

Another simplified approach to object tracking is to place fiducial markers under everyday physical objects so that they can be identified by cameras placed below the table surface. This has the advantage that everyday physical objects can be detected on a digital tabletop by simply placing stickers underneath them, thus wires and specialized hardware are not needed. Figure 2.10 demonstrates two systems that leverage fiducial markers over a digital table: reacTable (left) is a music synthesizer that uses the positions of various physical objects as controls on a digital table [Jordà *et al.*, 2007]. Recipe-table (middle) places fiducial markers below household cooking products to provide computer assisted cooking support [Kaltenbrunner and Bencina, 2007]. Recently, tangible tabletop support through fiducial markers has been made commercially available through the announcement of Microsoft Surface (www.microsoft.com/surface). Microsoft surface also provides other useful methods for tabletop interaction technologies such as multi-touch interaction and the capability of using pens, brushes and other markers to write on the display (similar to SmartBoards). Microsoft Surface is discussed in more detail in the next section.

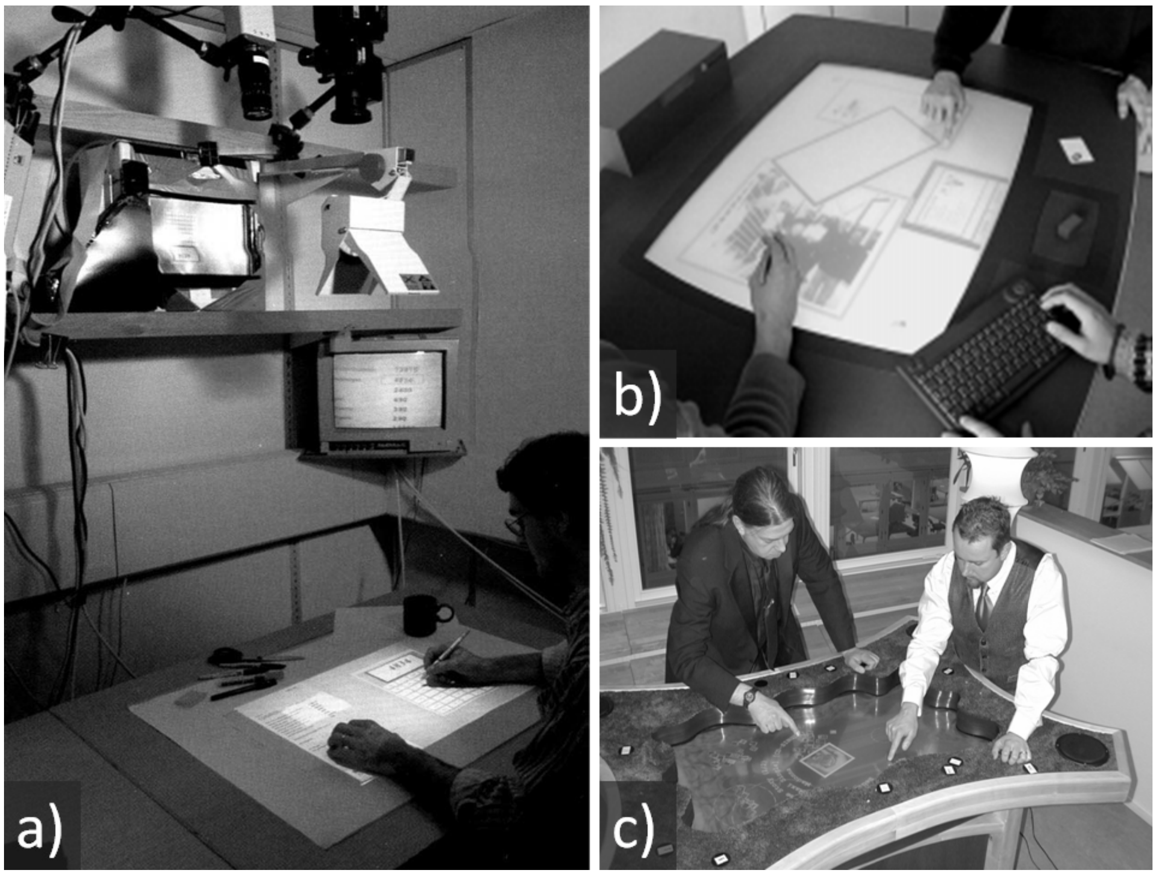


Figure 2.11. Early digital tabletop systems: Digital Desk by Wellner, 1991 (a), InteracTable by Streitz *et al.*, 1999 (b), The Pond by Ståhl *et al.*, 2002 (c)

Direct Touch. In everyday collaboration over physical tables people naturally use their arms and hands to move and reorient objects on physical tables. Instead of tracking the position of these physical objects as described earlier, some researchers have focused on tracking the movements of arms and hands as they come in contact with the table surface. As illustrated in Figure 2.11a, Digital Desk by Wellner [1991] tracked the position of a hand on a tabletop using an overhead camera. This system integrated paper with digital content as one could select numbers printed on sheets of paper and use it on in a digital calculator projected on a table. However, tracking the positions and movements of hands on the table was difficult and prone to error.

A simpler alternative to camera tracking is to place a touch sensitive overlay over the digital display to detect the location of touches on a tabletop surface. This approach was used in the InteracTable by Streitz *et al.* [1999] (Figure 2.11b) and The Pond by Ståhl *et al.* [2002] (Figure 2.11c), which both used a plasma display with a touch overlay. Both systems were limited to interaction with a single finger. If two simultaneous touches were used their values would be averaged placing the cursor halfway between the touches. This meant that the cursor would not appear near either of the collaborators' intended point of contact. This behaviour is frustrating because even accidental touches can cause the cursor to jump, thus people have to constantly coordinate their interactions with each other and the system [Ståhl *et al.*, 2002]. These accidental touches easily occur as people lean over and touch the table, as their clothing brushes the surface, or as their palms and elbows touch the surface.

A simple approach to multi-touch sensing is to use a plurality of discrete sensors, making an individual connection with each sensor. Individual capacitance sensors can be arranged in a matrix configuration to detect the position of touches over a table surface. This approach was used by Lee *et al.* [1985] to create a multi-touch sensitive tablet. It was extended in the commercial FingerWorks iGesturePad by Westerman et al [2001] to detect very small finger movements for a touch sensitive flat keyboard. iGesturePad supported multi-touch for typing as well as specific hand gestures (e.g. two fingers) for moving the mouse like the surface was a large touch sensitive mouse pad.

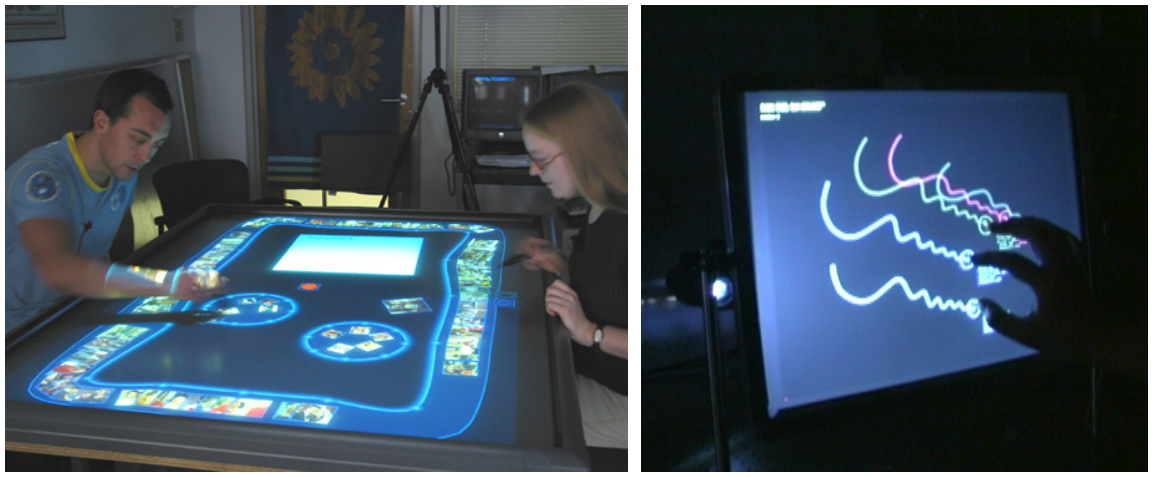


Figure 2.12. Infrared touch sensing technologies: DViT Smart Board, from Hinrichs *et al.*, 2005 (left) FTIR sensing, from Han, 2005 (right)

Another method uses infrared touch technologies to detect the heat signatures emitted or blocked by a touch on the tabletop. This too allows multiple points of contact to be detected. Multi touch detection is valuable as it expands the interaction possibilities from a single finger to multi finger and two handed interaction, it is also a requirement for multiple people to interact simultaneously. Hinrichs *et al.* [2005] used a multi touch DViT

SmartBoard horizontally as a digital table surface to support multiple simultaneous touches. Figure 2.12 (left) shows two people sorting photos using fingers or pens [Hinrichs *et al.*, 2005]. However as mentioned in §2.1.2, occlusion issues limit the number of simultaneous touches that can be DViT SmartBoard to two touches. This is problematic when there are three or more collaborators or when people want one wants to use multi-finger gestures. Another approach uses frustrated total internal reflectance (FTIR), an infrared sensing technique that allows an arbitrary number of touch points to be detected on a horizontal surface [Han, 2005]. FTIR is a well known technology that has been used in the biometrics community to image fingerprint ridges since at least the 1960s [White, 1965]. Earlier I described how FTIR technology emits infrared light from the side of the projection surfaces and uses a single camera underneath the surface to track multiple fingers. Since the camera is underneath it can also be used to view fudicial markers placed on the table surface. However, the reflected infrared light can be quite bright and difficult for the camera to detect.

Another approach to tracking tangible objects on a digital table is to emit infrared light from below to the touch surface and use multiple cameras to detect objects and fingers on the surface. TouchLight by Wilson (2004) used a large sheet of acrylic with two cameras placed behind the sensor and an infrared emitter located behind the surface to detect movements near the touch surface. This project was extended to detect fiducial markers on a digital table with the advent of Microsoft Surface. Since the fiducial markers are all illuminated equally on the surface even small fiducial markers can be detected. For instance, it is possible to see a fiducial marker etched onto a piece of clear glass. However, a limitation of both FTIR and rear IR tracking is that even though they can detect multiple touches, they cannot determine which touch belongs to whom.

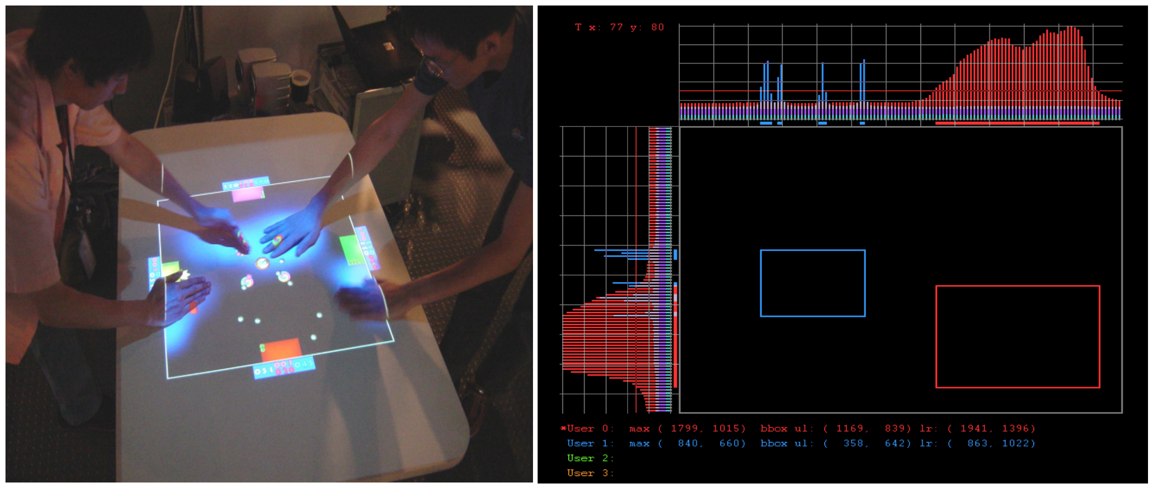
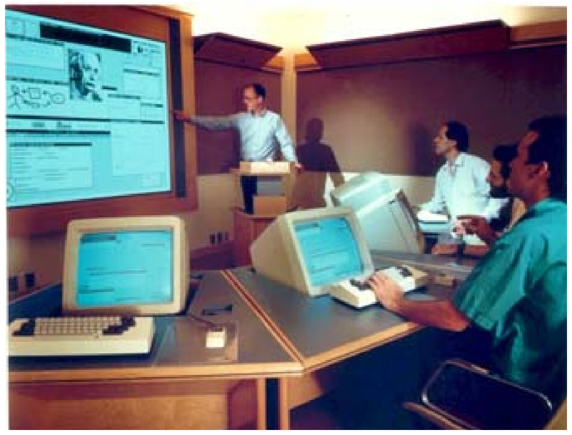

```
T x: 77 y: 80
*User 0:  max ( 1799, 1015)  bbox ul: ( 1169,  839) lr: ( 1941, 1396)
 User 1:  max (  840,  660)  bbox ul: (  358,  642) lr: (  863, 1022)
 User 2:
 User 3:
```

Figure 2.13. Capacitive Touch Technologies: SmartSkin, from Rekimoto, 2002 (left) Input visualization of two people touching the DiamondTouch, from Dietz and Leigh, 2001(right)

Next, capacitive touch technologies provide multi-touch input that can be used to infer hand postures and also provided information about the person who made each touch. This is important as multi finger and whole hand postures allow collaborators to use more expressive gestures when interacting over a digital table. Touch identification is useful for detecting hand postures from an individual that involve multiple fingers such as a five finger grabbing gesture. Researchers also benefit from touch identification in when logging the activities of multiple people on a digital table. This data can be used to inform researchers of how people use digital tables in practice.

As shown in Figure 2.13 (left) SmartSkin by Rekimoto [2002] was able to detect multiple points of contact (shown by the halo around each person's hand) but was not able to identify who made each touch. The Mitsubishi Electric Research Labs (MERL) DiamondTouch provided user identification with each touch [Dietz and Leigh, 2001] but was not able to uniquely identify multiple points of contact from the same person. Despite this limitation, Wu and Balakrishnan [2003] demonstrated several multi finger and whole hand posture recognition methods over a MERL DiamondTouch table, including one finger, one hand, one hand side, two fingers, two hand sides, and two corner shaped hands. Since infrared technologies can detect the size and properties of the hand in contact with the surface they should also be able to detect various hand postures like those presented by Wu and Balakrishnan [2003].

These capacitive touch technologies use a matrix of horizontal and vertical antennas to sense where multiple people are touching on a large display. A frequency is emitted from each antenna in sequence. This frequency modifies the capacitance of a person's body as they touch the display. People sit on capacitive pads that listen to this change in capacitance. A visualization of the input of two people interacting with the digital table is shown in Figure 2.13 (right), one person is using five fingers (blue) and the other person is using one arm (red). The signal strength of each vertical and horizontal antenna is the sum of all touches on that wire, thus providing useful information for detecting the hand postures of multiple people. This is further discussed in Chapter 5.



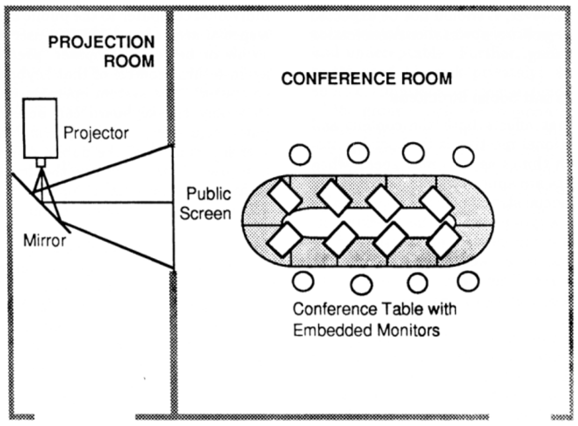


Figure 2.14. Early networked computer environments: Co-lab, from Stefik *et al.*, 1987 (left) CaptureLab, from Elwart-Keys *et al.*, 1990 (right)

2.1.4 Interactive Surface Environments

Groups often collaborate over a variety of surfaces rather than just a single one. Consequently, researchers are exploring environments equipped with multiple digital surfaces to support collaboration. I begin with a discussion of networked computer environments, I then discuss public item transfer when using PCs and PDAs with large displays,

Networked computers. Early interactive surface environments were developed to support typical meeting room activities by providing each collaborator with a computer and connecting them via a network to a large, shared vertical display (e.g. Stefik *et al.*, 1987, Elwart-Keys *et al.*, 1990). Figure 2.14 illustrates two networked computer environments: in

Co-lab (left) participants were seated in a semi circle facing a shared vertical display [Stefik *et al.*, 1987] whereas in CaptureLab (right) participants were seated perpendicular to the shared vertical display. Co-lab was a multi user system that understood the inputs of multiple people, whereas in CaptureLab people were all sharing the same single user system.

Using personal computers as input to a large display has problems. Cognoter was a brainstorming application for Co-lab that enabled people to write notes on individual computers and share them on a large vertical display. Studies of this system revealed that people had a tendency to focus on their own display rather than looking at the large shared display. They lacked awareness of other's actions because they could not see what others were working on until they submitted their notes to the shared display [Tartar *et al.*, 1991]. This is typical of interactions with a mouse and keyboard as these small motions are difficult for others to see and interpret as awareness information [Gutwin and Greenberg, 2004].

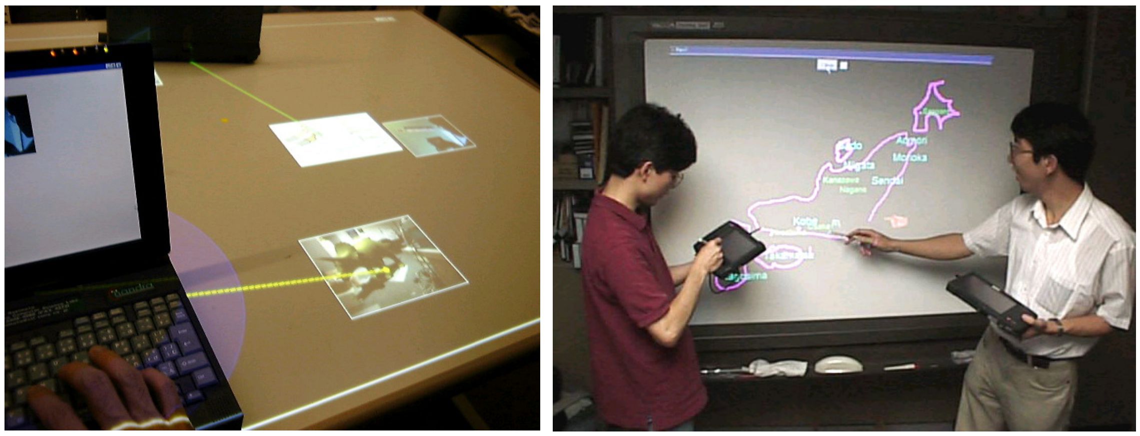


Figure 2.15. Making transfer actions publicly visible in Augmented Surfaces, from Rekimoto and Saitoh, 1999 (left) and Pick-and-Drop, from Rekimoto, 1997 (right)

Public item transfer when using PCs and PDAs with large displays. Awareness can be augmented by making digital object creation and manipulation publicly perceptible through visualization and gesture. In Cognoter, people had difficulty connecting notes on the vertical display to the computers used to create it. This made it difficult to clearly perceive the contributions of other collaborators. To improve awareness of information flow between personal computers and public displays Rekimoto and Saitoh [1999] placed laptops on a top projected digital table and visualized the connection between the laptop and the digital objects being manipulated using a yellow line. As illustrated in Figure 2.15 (left) the yellow line appears depicting the transfer of an image from a laptop to the digital table. This produces consequential communications that others can use for awareness.

Awareness can also be improved by using hand gestures to transfer information between two computers. For example, Rekimoto [1997] introduced the pick and drop technique that allowed multiple collaborators to move items from a Tablet to a large vertical display. As illustrated on Figure 2.15 (right), object transfer started with a touch on a tablet computer specifying the object to transfer followed by a touch on a large vertical display specifying the destination location. This means that an explicit visualization of object movement is not needed. Instead this technique leverages what people naturally do in collaborative environments.

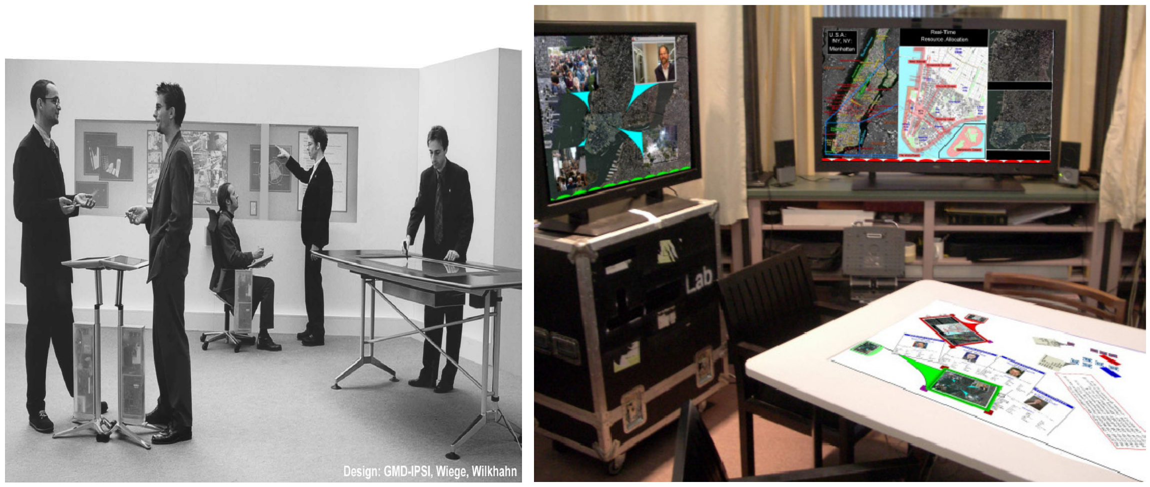


Figure 2.16. Smart rooms: i-Land, from Streitz *et al.*, 1999 (left) Tabletop world in miniature views, from Wigdor *et al.*, 2006 (right)

Smart rooms. Smart rooms provide collaborators with a variety of displays including large public wall and table displays as well as small personal computers and tablets. These displays are connected with a networked architecture that allows information to move seamlessly across displays.

The Interactive landscape (i-Land) shown in Figure 2.16 (left) allowed images and documents to be transferred across surfaces through wooden blocks equipped with radio frequency identification (RFID) tags that acted as virtual object containers [Streitz *et al.*, 1999]. Each display included an RFID reader which would act as a virtual folder when a tag was placed on it. Collaborators could move items to the virtual folder to "store" them on the tag and then they could "retrieve" these items on a separate display using by moving the tag and placing it on a different RFID reader. No information is actually stored on the tags. Instead, RFID numbers are tracked by the networking architecture and files are stored on local hard drives.

When working over multiple displays there may be a need to manipulate artefacts across several displays and computers. A simple approach is to provide a VNC like architecture for controlling multiple displays using a keyboard and a mouse. Booth *et al.* [2002] developed the Mighty Mouse system that allowed a single keyboard and mouse to control several different displays and computers in a co-located setting. They provided floor control mechanisms to handle multiple people interacting simultaneously.

Another more public approach is to show miniature versions of distant displays on a digital table. Wigdor, et al [2006] provided world-in-miniature views of peripheral displays to allow the digital table to control actions on peripheral vertical displays. As seen in Figure 2.16 (right), items could be moved from the table to peripheral vertical displays by dragging them from the digital table to the world in miniature views (the coloured boxes on the digital table) representing the peripheral displays. Another approach is to point to peripheral displays using a hand or laser pointer. The Perspective cursor by Nacenta *et al.* [2006] used a magnetically tracked stylus to move a cursor across multiple displays similar to using a laser pointer.

2.2 Multimodal Interaction Technology

In this section, I discuss multimodal interaction technologies within the context of the co-located interaction. When people work together they often need to coordinate their activities with others. Gutwin and Greenberg [2004] argue that awareness of other's activities can facilitate this coordination. Multimodal interaction technology can facilitate activity awareness by making the actions of individuals more perceptible to collaborators. Thus my focus is on explicit publicly perceptible speech and gesture commands rather than implicit, ambient of perceptual multimodal interfaces. I begin with a discussion of early gesture and speech recognition technologies leading up to the late 1990s, which describes the state of the art in readily accessible recognition technologies at the time of this writing. I then discuss systems that fuse gesture and speech input over a variety of technology environments. I describe desktop computers augmented with multimodal interaction. Then I discuss how multimodal interaction has been deployed over large digital displays. I then detail the issue of multimodal saliency (i.e. knowing when multimodal commands should be directed to the computer versus posture adjustments and those used in regular communication). Finally, I discuss multimodal technology advances in virtual and augmented reality environments.

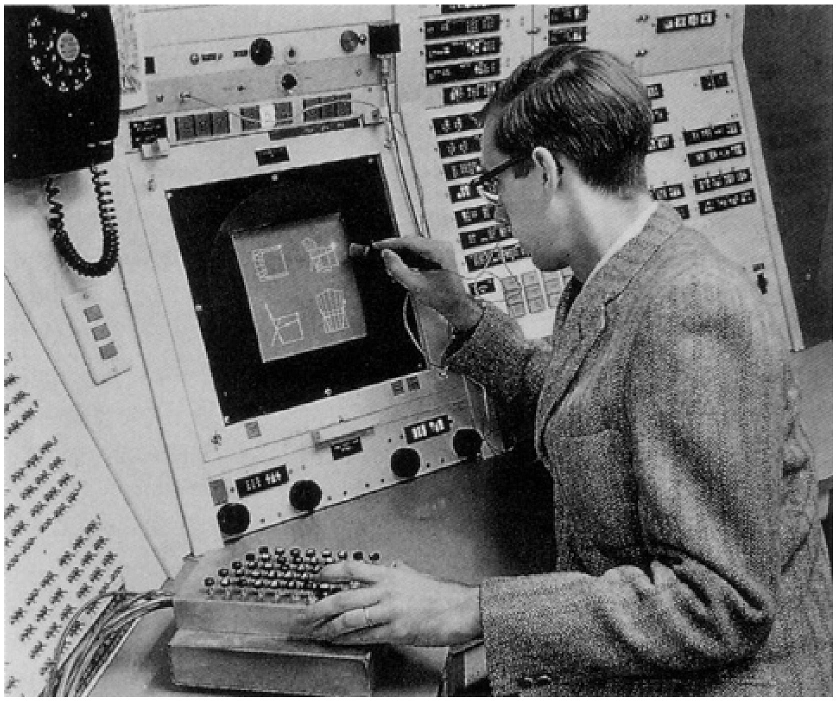


Figure 2.17. The Sketchpad interactive drawing application from Sutherland [1964]

2.2.1 Early Gesture Recognition

Many of the large display gesture recognition systems discussed earlier expand on an extensive history of gesture recognition research. Gestures over pen based tablets have been explored since the mid 1960s. The SketchPad system by Sutherland [1964] explored the use of gesture recognition systems by tracking the position of a light pen on a tablet computer. Gesture recognizers monitored rapid movements such as a rapid flick to indicate that a drawing action was completed. This system introduced many researchers to concepts such as direct manipulation with graphical objects, a precursor to the graphical user interfaces later demonstrated in the Alto by Xerox's Palo Alto Research Center (PARC) in 1974.

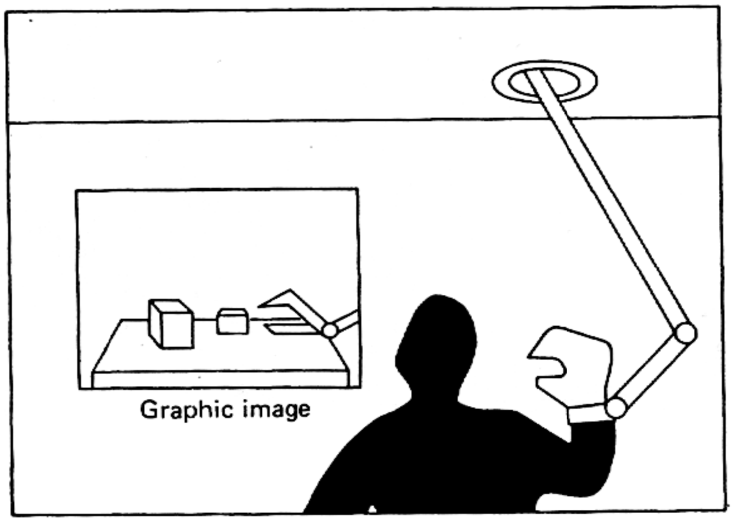


Figure 2.18. An illustration of the Grope System by Kilpatrick [1976]. Illustration from Krueger [1991].

Gesture recognition had been deployed to commercial CAD systems in the 1970's with pen-based tablet systems being sold commercially by Wacom in the early 1980s. Such systems were generally designed for a single individual working over a desktop computer. At the same time gesture recognition technologies were also being extended to systems that could track an individual's hand, gaze and body movements. For example, Kilpatrick [1976] developed the Grope system, which connected a mechanical radioisotope manipulator arm to a simple graphical world as illustrated in Figure 2.18. Alternate sensing technologies such as magnetic and acoustic sensing technologies [Sturman and Zeltzer, 1994] are less cumbersome than a mechanical arm but they still require the user to wear external sensors. This can make it difficult to manipulate objects in the physical environment.

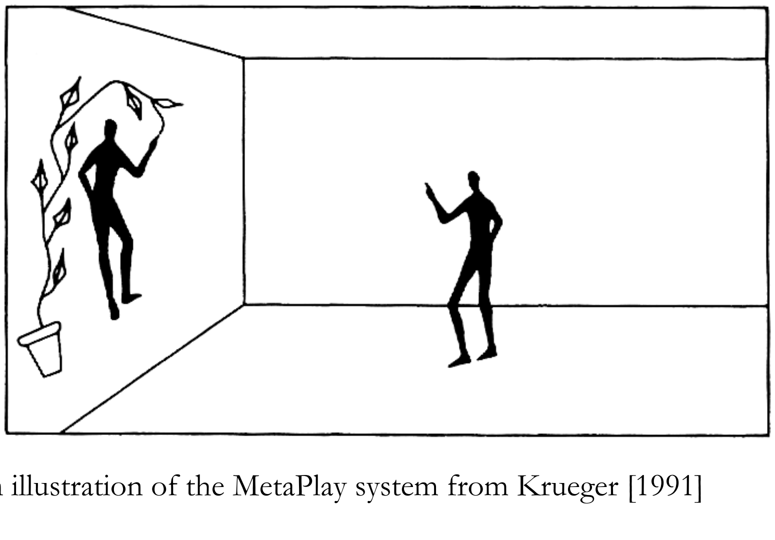


Figure 2.19. An illustration of the MetaPlay system from Krueger [1991]

An alternative approach is to use computer vision techniques to track the hands and body without the use of external sensors. Camera based tracking is sensitive to ambient lighting conditions, thus researchers have use various lighting techniques to track the hand and body. An early example includes MetaPlay by Krueger, originally demonstrated in 1970. As seen in Figure 2.19, a person's silhouette would be projected onto a wall using a light source behind the person. MetaPlay was able to track the movements of the body by tracking the shadow of a person standing in a large room [Krueger, 1991]. These silhouettes were tracked by an overhead camera that would allow a person to interact with a rear projected display. For example in Figure 2.19, a person is interacting with a digital tree using their shadow from a distance. One important aspect of this work was that it involved multiple hands and full body interaction. For example, in the VideoPlace system a person could swing their arms to interact with a virtual critter hovering around the person's silhouette [Krueger, 1991].

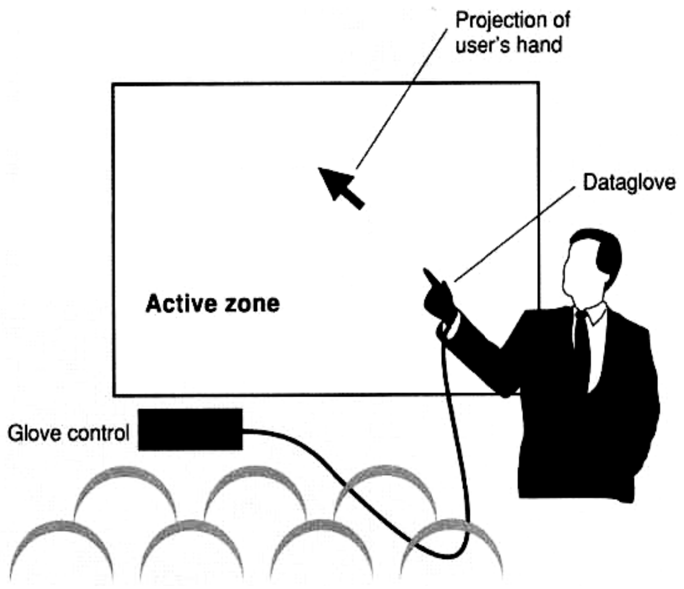


Figure 2.20. An illustration of the active zones in the Charade System from Baudel and Beaudouin-Lafon [1993]

One issue that arises when using always-on tracking systems is that the computer is unable to determine when actions are intended for the computer versus when they are postural adjustments or intended as communication to collaborators. This is commonly referred to as the ***gesture saliency problem***. The Charade system by Baudel and Beaudouin-

Lafon [1993] provided active and inactive regions for gestures. Gestures within the active region would perform actions on the system while gestures in the inactive regions are treated as posture adjustments or communication to others. This approach can result in errors if people are not paying attention to the position of their hands when gesturing. An alternative is to use a physical surface such as a wall or table to perform gestures intended for a computer. For example, Sketchpad [Sutherland, 1964] only recognized gestures when the light pen was placed on the display surface; any gestures outside of the surface were not recognized by the system. As we will see in Chapter 3, I also use this approach to detect hand postures on a digital table.

By the end of the 1990s gesture recognition had been deployed for widespread use in portable technologies such as Personal Digital Assistants (PDAs), Tablet Personal Computers (TabletPCs), mobile phones, and Portable Media Players (PMPs). Many people are now familiar with gesture recognition technologies and expect to use it in their everyday work practices.

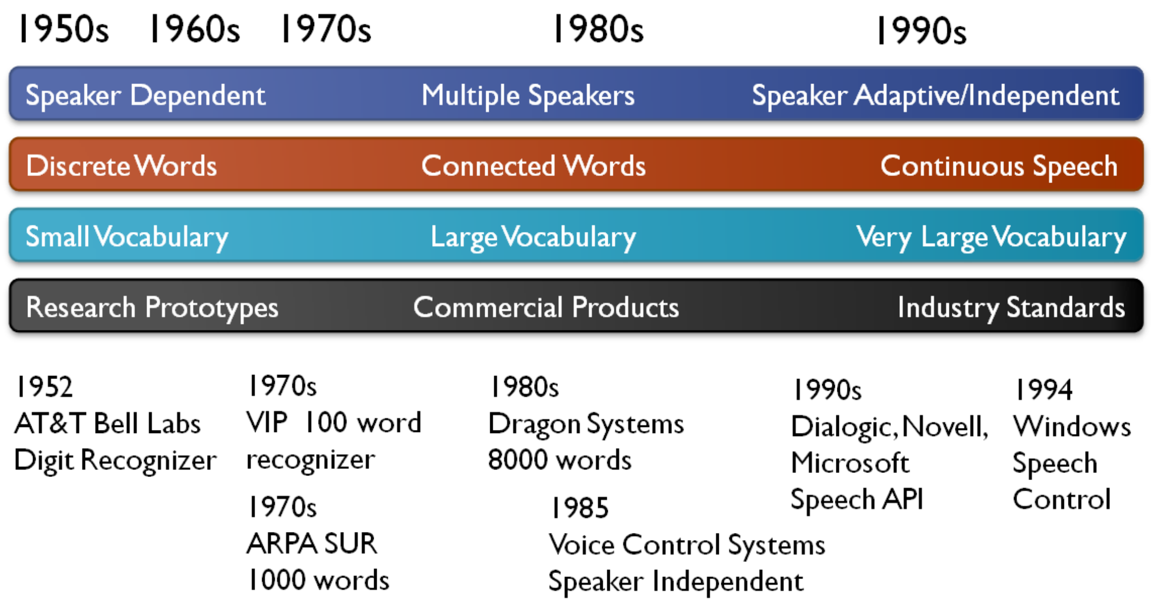

1950s
1960s
1970s
1980s
1990s
Speaker Dependent
Multiple Speakers
Speaker Adaptive/Independent
Discrete Words
Connected Words
Continuous Speech
Small Vocabulary
Large Vocabulary
Very Large Vocabulary
Research Prototypes
Commercial Products
Industry Standards
1952 AT&T Bell Labs Digit Recognizer
1970s VIP 100 word recognizer
1970s ARPA SUR 1000 words
1980s Dragon Systems 8000 words
1985 Voice Control Systems Speaker Independent
1990s Dialogic, Novell, Microsoft Speech API
1994 Windows Speech Control

Figure 2.21. A timeline of early speech recognition technology

2.2.2 Early Speech Recognition

This section provides a brief history of the evolution of speech recognition technology and is illustrated through a visual timeline in Figure 2.21. This visual timeline illustrates advances in speech recognition technology areas such as speaker dependency (i.e. the speaker must be trained on the system), discrete/continuous wording (i.e. are pauses required between words), vocabularies ranging from 10 words to over 30,000 words, and the transition from research prototypes to industry standards. This summary is an expanded and rephrased from the speech recognition history found in Using Speech Recognition by Markowitz [1996].

Speech recognition history begins in the late 1930s, when Dudley of AT&T Bell Laboratories proposed a system model for speech analysis and synthesis [Dudley, 1939]. However, it was not until 1952 when Davis et al., from AT&T Bell Laboratories demonstrated the first machine capable of recognizing speech. This system recognized ten English digits by comparing stored templates to those spoken through a microphone [Davis *et al.*, 1952]. This brute force approach required extensive tuning to recognize the speech of a single dependent speaker. By the mid 1960s it was clear to most researchers that speech recognition was a significantly nuanced problem. Consequently, researchers narrowed their focus to systems capable of handling the speech of one person (speaker dependent), inputs containing pauses between words (discrete-word speech), and vocabularies of 50 words or fewer (small vocabulary systems).

The first commercial speech recognition products were developed in the early 1970s. The VIP 100 system of Threshold Technology, Inc. demonstrated a speaker dependent, discrete-word, small vocabulary speech recognizer. This system demonstrated the viability of speech recognition, and generated significant interest from the United States Department of Defense and their funding body: the Advanced Research Projects Agency (ARPA). ARPA subsequently developed the Speech Understanding Research (SUR) program that drove speech recognition research towards multiple speaker dependent recognition, connected word speech, and vocabularies of 1000 words or more.

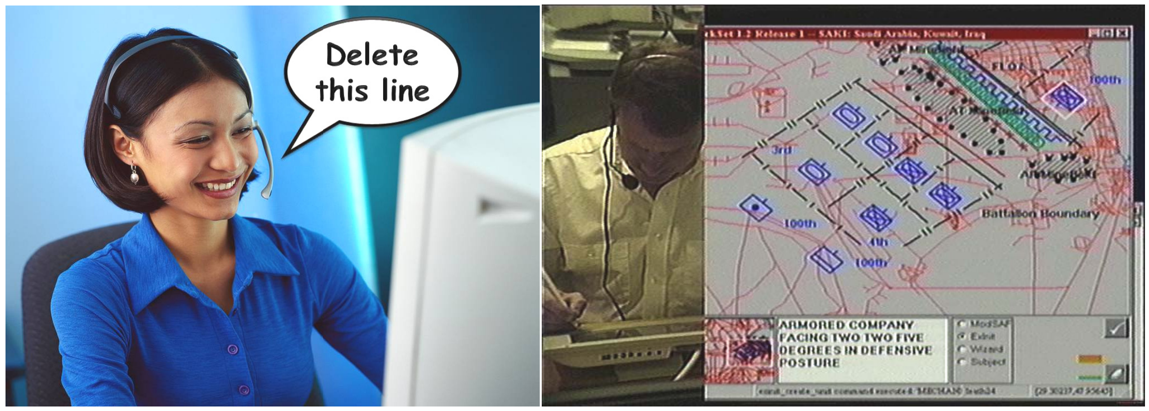
Commercial speech recognition systems with continuous speech recognition capability began to appear in the 1980s. Dragon Systems introduced a speaker adaptive (i.e. could be trained), discrete-word dictation system that boasted a vocabulary of 8,000 words. Also, significant advanced in noise reduction technology made it feasible to develop speech recognition systems over a telephone. The first speaker independent cellular telephone dialling system was demonstrated by Voice Control Systems in 1985. Any speaker could use their cell phone dialling system without any prior training.

By the 1990s, numerous commercial applications for speech recognition had emerged. For example, in 1997 Dragon System launched the Dragon NaturallySpeaking product, a continuous speech recognition system for general purpose use with a vocabulary of 23,000 words. In the same year, IBM offered its own continuous speech recognition system called ViaVoice. In the first year alone these products sold over 75,000 copies. Dialogic, Novell, and Microsoft later sponsored efforts to develop a standard Application Programmer's Interface (API) for speech recognition systems.

Prior to 1990, people's experience with speech recognition technology was largely limited to the works of science fiction (e.g. Star Trek). By the late 1990s many people had either used or heard of speech recognition systems in a variety of application domains such as controlling Microsoft windows interfaces, voice activated dialling, and commercial telephone response systems. Speech recognition was changing from novelty to a tool for everyday use.

2.2.3 Multimodal Fusion

While gesture recognition and speech recognition technologies have a long history dating back to at least the 1950s, the combination of both technologies is a much more recent affair. Bolt's Put-That-There multimodal system from the early 1980s is considered one of the first systems to fuse the input of a magnetically tracked gesture recognizer and a speech recognition engine as a unified stream of input to a computer system [Bolt, 1980]. This



ARMORED COMPANY FACING TWO TWO FIVE DEGREES IN DEFENSIVE POSTURE

Figure 2.22. Multimodal desktop extensions: An illustration of IBM's Human Centric Word Processor [Papineni *et al.*, 1997] (left) Interaction with a tablet computer in Quickset, from Cohen *et al.*, 1998 (right)

system accepted all speech and gesture actions as inputs to the computer. Cohen *et al.* [1997] published a multimodal fusion technique to help guard against false speech or gesture recognitions, illustrated in Figure 2.22 (right). For example, the system would know that a "move here [point]" command requires both a speech and gesture components. If only the speech component was recognized within a certain time threshold, this would be treated as an erroneous recognition since no corresponding gesture was provided. In 2003, Kaiser *et al.* presented a statistical multimodal fusion technique incorporating speech, gesture, and gaze input. This technique would combine the inputs of three different input modalities: speech, gesture and gaze and would use the n-best recognition hypotheses to produce a more accurate integration result.

2.2.4 Desktop Extensions

I described in Chapter 1 how individual actions over a personal computer are inherently difficult for others to perceive due to the small motions of the hands when using a keyboard and mouse. A simple way to make actions publicly perceptible over desktop computers is to use speech recognition technology to interact with applications. While this has not been the focus of multimodal desktop systems, it is a desirable side effect for collaborative situations.

For example, IBM's Human-Centric Word Processor allowed an individual to make modifications to a text document using speech commands and a mouse [Papineni *et al.*, 1997]. This system was designed to assist the user in making corrections after they had dictated text using a speech recognizer. As illustrated in Figure 2.22 (left), a person could select a line of text with a mouse and say "delete this line" to remove it. Similarly, to move a line of text one could use the multimodal command "move this *[select a sentence]* here *[point to location]*".

Pen and voice with tablets. To provide additional gestural awareness one can exploit direct manipulation using a pen over a digital tablet coupled with speech interaction. I caution that the systems described below were not designed for awareness in collaborative situations. Quickset by Cohen *et al.* [1997] supported pen and voice interaction over a digital map for military command and control scenarios. People could position units on a map by pointing to a location with the pen and saying "create armoured company" (Figure 2.22, right).

Similarly, they could position and orient units with a multimodal command (e.g. "*[select unit]* facing 225° in defensive posture"). Speech and gesture commands had to be unified to be accepted as input in Quickset. For example the "create unit here *[point to location]*" would require both speech and gesture components. If only the speech component was recognized within a certain time limit then the speech command would be ignored and treated as an erroneous recognition. Studies of this system have been shown to provide error rate reductions of 19-40% compared to multimodal systems that wait for multimodal input to be fused together [Oviatt, 1999].

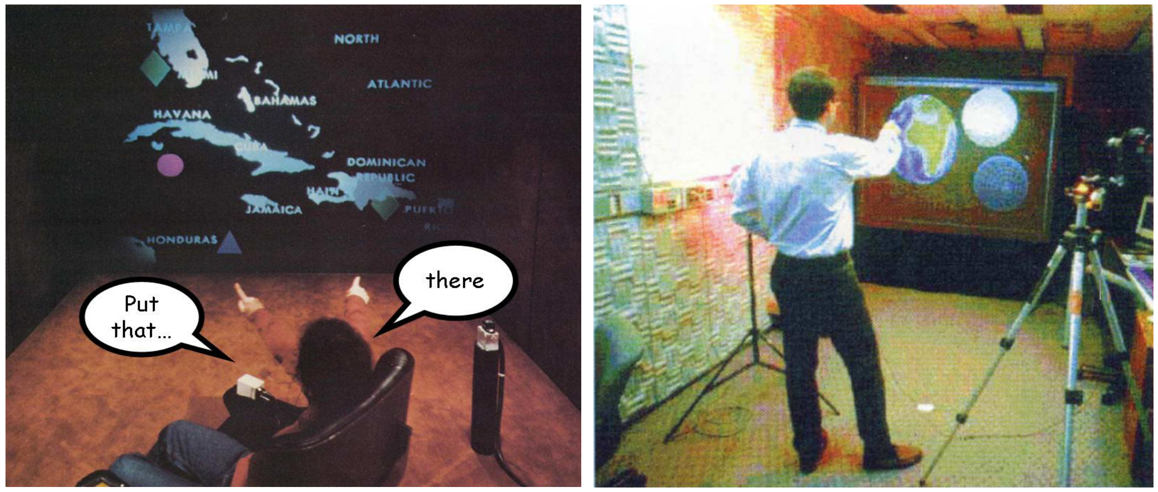


Figure 2.23. Multimodal interaction over wall displays: Put-that-There, modified from Bolt, 1980, camera based hand tracking from Billinghurst, 1998

2.2.5 Multimodal Interaction with Large Displays

Multimodal interaction with wall displays. Gestures performed over large digital displays require larger arm movements and can produce more activity awareness. Early large display gestural interaction includes Krueger's [1977] VideoPlace. This system used an individual's silhouette to interact with virtual objects a large display. Similarly, one of the earliest examples of multimodal interaction involved moving objects over a large wall display. As shown in Figure 2.23 (left), Bolt's [1980] seminal Put-That-There system illustrated the use of speech and gesture for distant freehand selection over a large wall display. It was the first system to demonstrate the unification of speech and gesture recognition technologies for interaction with a computer system. A person could manipulate targets on a digital map using the multimodal command "put that *[points to item]* there *[points to location]*". This multimodal approach provides individuals with a brief and simple method of selection that mimics how one might instruct another to move items on a wall. An unintended side effect of this approach is that these actions are easily understood by others during collaborative work.

Bolt's Put-That-There system [1980] inspired many future replications using alternative input technologies. Some have used low cost vision based tracking [Billinghurst, 1998] to mimic similar interactions. Others have used alternate input devices to explore novel multimodal interactions [Corradini *et al.*, 2003].

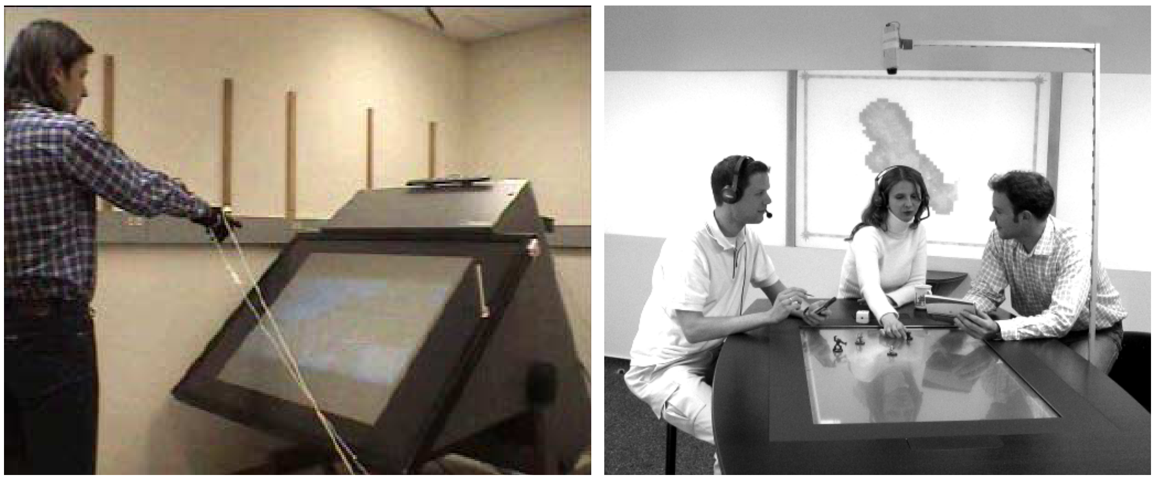


Figure 2.24. Alternative display form factors for multimodal interaction: a drafting board orientation from Corradini *et al.*, 2002, a tabletop surface from Magerkurth *et al.*, 2004.

Multimodal interaction over drafting board and tabletop oriented displays. Alternate large display form factors have been explored for multimodal interaction. For example, Figure 2.24 (left) shows how Corradini *et al.* [2002] explored multimodal interaction over a drafting board sized display using speech and 3D gestures. In addition to simple pointing gestures this system could also detect the twisting of a hand to engage panning on a digital map. While this gesture expands the possible multimodal interaction over large displays, it is subtle and can be easily missed by others.

Digital tables facilitate face-to-face communication as people can see both the table contents and the gestures/expressions of others simultaneously. Magerkurth *et al.* [2004] implemented a multimodal tabletop system in the Interactive Landscape smart room described in §2.1.4. People could move physical game pieces (tracked by an overhead camera) and issue speech commands in a turn based board game. Due to technological limitations this system could not support multiple simultaneous speech or gesture actions, thus people had to explicitly take turns to interact with the system. Nonetheless, this is likely the earliest example of multimodal interaction over a digital table.

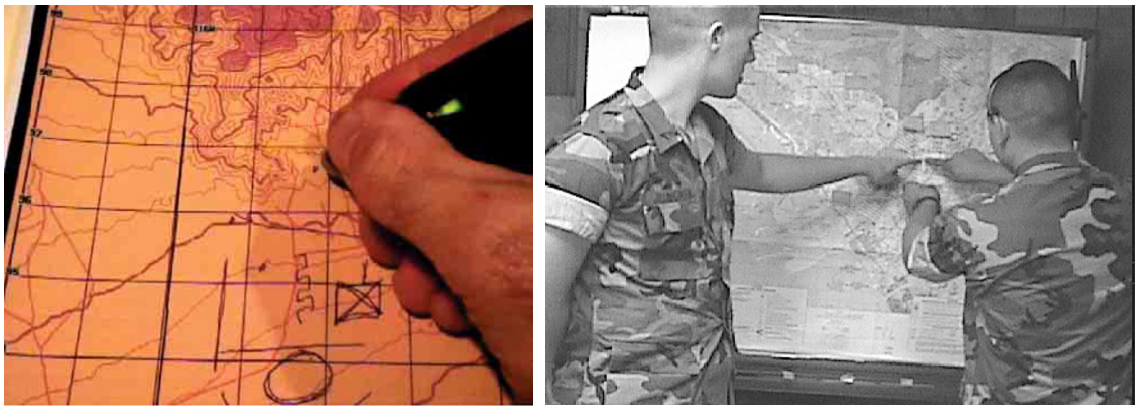


Figure 2.25. Multimodal tangible interaction: with a digital pen in NIS Map, from Cohen and McGee, 2004 (left) with a SmartBoard in RASA, from McGee and Cohen, 2001 (right)

Multimodal interaction with paper. As described in Chapter 1, paper is heavily used in highly collaborative work such as military command and control. Researchers have applied this knowledge to the design of paper based multimodal systems. This has the benefit that collaborators need not change their existing work practices to use a multimodal system. McGee and Cohen have done extensive research on multimodal input over paper media for military command and control situations. NIS Map by Cohen and McGee [2004] used a headset microphone and an Anoto digital pen (containing a camera used over position coded paper as in §2.1.3) to capture the speech and pen based gesturing on a paper map so that it could be later added to a digital system (Figure 2.25, left). This hybrid approach combines people's collaborative work practice with the power of digital systems. For example, the digital system can be used for logging, conflict detection and calculation.

People also use sticky notes to represent dynamic content on physical maps as these notes can be easily repositioned. In collaborative command and control situations, it is common to use sticky notes to denote ground troops in strategic planning. It is possible to monitor the movement of sticky notes by placing a touch sensitive SmartBoard behind the paper map and using an overhead camera to read the note contents. This was done by the RASA system by McGee and Cohen [2001]. As illustrated in Figure 2.25 (right), the system could monitor the dynamic manipulations of sticky notes over a paper map so that it could be later used with a digital system.

2.2.6 Multimodal Saliency

One issue that arises when using multimodal interaction in a multi user setting is that always on speech recognition and gesture recognition systems need to determine when a person is intended for a gesture/speech action is to be recognized by the computer and when it is meant as conversation to others. For gestures, one can specify a particular region of the collaborative space to accept commands from the computer (as mentioned in §2.2.1). For speech, computers have great difficulty identifying the appropriate interlocutor as people use many nuanced and subtle actions to direct attention. In fact, most commercial speech recognition systems instruct users to turn off the microphone if they are not speaking to the computer. Recognizing the importance of this issue, researchers have explored approaches for detecting saliency in multimodal systems. For example, a Wizard of Oz study conducted by Lunsford *et al.* [2005] examined the differences in people's behaviours when speaking to a computer versus to other people. They discovered that people of median age spoke to the computer louder, and more articulate than when speaking to another collaborator. Their findings indicate that a volume configuration (with appropriate age adjustments) could help a computer know when speech was meant as a command.

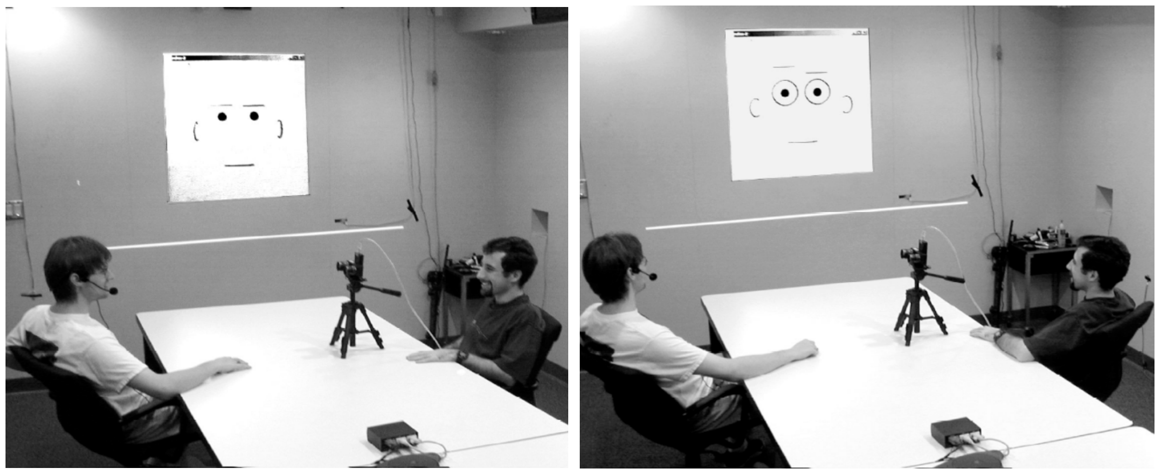


Figure 2.26. Look-to-talk multimodal interaction: looking at another person (left) looking at the computer (right).

Another complementary approach has been to leverage the gaze modality to determine when speech is meant to be directed to a computer. Oh et al. [2002] explored the

Look-To-Talk interface that determined where a person was looking and only used spoken phrases when looking at the computer. An evaluation of this system found that users rated Look-To-Talk significantly higher in terms of naturalness that a Push-To-Talk alternative. Such systems could be combined with other modalities and techniques to improve a computer's perception of multimodal saliency.

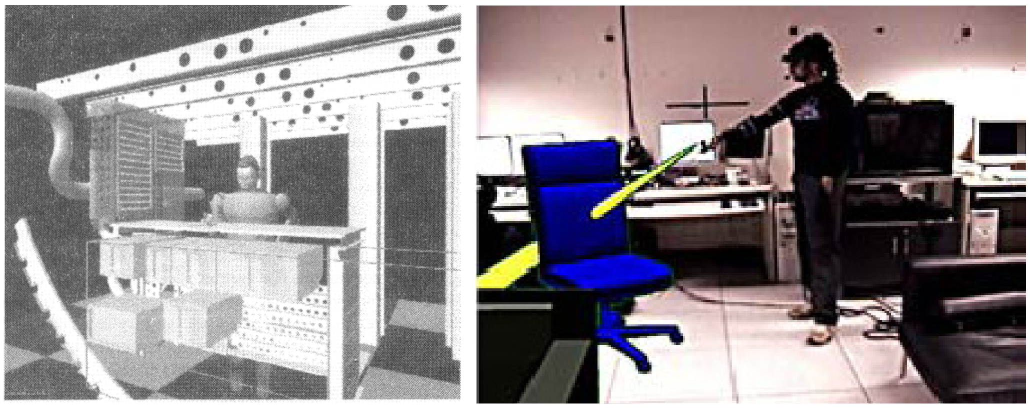


Figure 2.27. Multimodal interaction in virtual and augmented reality environments: Boeing's VR Aircraft Maintenance, from Duncan *et al.*, 1999, MAVEN from Kaiser *et al.,* 2003 (right)

2.2.7 Virtual and Augmented Reality

Multimodal interaction has been used in many virtual, mixed and augmented reality environments where users wear head mounted displays (HMDs) to see digital content. In virtual reality (VR) systems, the user is immersed in an artificial 3D world generated by a computer where he or she typically controls a virtual avatar corresponding to the motions of that person in the physical world. Since these 3D worlds are generated on the computer, they need not be constrained to the limits of physical reality. For example Figure 2.27 (left) shows Boeing's VR Aircraft Maintenance application. A person can say "fly forward" to reposition themselves or say "take me to the E4 table rack" to fly to that location. This allows the maintenance trainee to focus on the task of learning how to fix the aircraft rather than working around the logistics of travelling from within a virtual airplane.

Augmented reality (AR) systems can be used to highlight actions that might otherwise be missed during collaborative work. One approach is to overlay 3D images on top of a collaborator's gestures thus making it clear where a person is gesturing. This was done in the MAVEN system by Kaiser *et al.* [2003]. As shown in Figure 2.27 (right), pointing gestures were augmented with a 3D cone shape emanating from a person's hand. Thus another collaborator wearing a HMD could see the yellow cone to get a better sense of the object that someone was pointing to. MAVEN advanced multimodal recognition by fusing the speech, gesture and gaze recognition results to probabilistically determine an individual's target object. A study of this system revealed that using the multimodal inputs produces a more robust system than would be possible with the recognition success of a single modality [Kaiser *et al.,* 2003].

While the augmented and virtual reality systems described above focused on individual speech, gesture and gaze actions, its contributions have implications for multi user systems as well as collaborators can leverage location flying and visually augmented gestures in their collaborative work practices.

2.3 Conclusion

In this chapter I provided background regarding previous technical explorations in co-located and multimodal interaction. In the co-located section, I described Single Display Groupware systems that leverage multiple mice and keyboards. I then described technologies for wall and table displays. Finally, I reviewed technologies for interactive surface environments. Each technology has certain benefits and tradeoffs and their suitability for collaborative work will depend heavily on the situation and tasks people need to perform.

In the multimodal section I focused specifically on technologies that can produce actions publicly visible to others. This section began with a discussion of speech and gesture recognition technology and the fusion of these two input modalities. I then discussed multimodal desktop extensions and multimodal interaction has been used over large digital displays. Finally, I discussed new interaction possibilities in virtual and augmented reality systems. I stress that multi user interaction has not been the focus of these systems. However, the numerous figures featuring multiple collaborators indicate that there is a strong interest in supporting group work.

The focus of this thesis is not to advance co-located or multimodal technology. I emphasize that this thesis extends this prior work described in this chapter by combining both multimodal interaction and co-located interaction. As seen in this review, this topic has been barely touched on by previous research. In the following chapter I motivate this combination of co-located and multimodal technology by examining the theoretical, empirical and ethnographic research on people's collaborative work practices.

Chapter 3. Motivating Interaction with Single User Applications through Speech and Gestures on a Multi-User Tabletop

Objective One. *I will distill existing theories, empirical and ethnographic studies into a set of behavioural foundations that inform the design of multimodal co-located systems and outline individual and group benefits.*

Objective Two. *I will develop collaboration-transparent multimodal co-located wrappers over existing applications.*

In Chapter 2, I explored technologies that support co-located work and multimodal technologies whose actions are publicly perceptible to others. Yet none of this research has focused on the combination of co-located and multimodal technology. In this chapter, I motivate speech and gesture interaction for co-located cooperative work. By synthesizing a number of theories, empirical and ethnographic studies on people's collaborative work practices I establish a set of behavioural foundations that summarize individual and group benefits for multimodal interaction. The behavioural foundations establish that people use a combination of speech and expressive hand gestures during collaborative work.

With the advent of large touch sensitive tables and speech recognition, we can recognize some of the expressive gesture and speech actions that people use in collaborative work and apply them to digital table design. However, researchers are limited by the difficulty of building a truly useful collaborative application from the ground up. I circumvent this difficulty by wrapping existing off-the-shelf single-user applications with a multimodal interface. By using existing applications I can focus on the design of multimodal interfaces that serve dual purpose as commands to the computer and as communication to others. Through three case studies of single user applications I show the new functionalities, feasibility and limitations of leveraging such single-user applications within a multi user, multimodal tabletop.

3.1 Introduction

While there are many factors promoting information exploration on physical tables versus desktop computers (e.g., insufficient screen real estate and low image resolution of monitors), an often overlooked problem is that *personal* computer systems are designed within single-user constraints. Only one person can easily see and interact with information at a given time. Although multiple people can work on the same system by turn-taking, the system is blind to this fact. Even if a large high resolution display is available, one person's standard window/icon/mouse interaction – optimized for small screens and individual performance – becomes awkward and hard to see and comprehend by others involved in the collaboration [Segal, 1994]. [1]

For a computer system to be effective in such collaborative situations, the group needs at least: (a) a large and convenient display surface, (b) input methods that are aware of multiple people, and (c) input methods that leverage how people interact and communicate over the surface via gestures and verbal utterances [Cohen *et al.*, 2002, Oviatt, 1997]. For point (a), I argue that a ***digital tabletop display*** is a conducive form factor for collaboration since it lets people easily position themselves in a variety of collaborative postures (side by side, catty-corner, round table, etc.) while giving all equal and simultaneous opportunity to reach into and interact over the surface. For points (b) and (c), I argue that ***multimodal gesture and speech input*** benefits collaborative tabletop interaction.

[1] Portions of this Chapter are published as:

Tse, E., Shen, C., Greenberg, S. and Forlines, C. (2006) Enabling Interaction with Single User Applications through Speech and Gestures on a Multi-User Tabletop. Proceedings of Advanced Visual Interfaces (AVI'06), May 23-26, 336-343, Venezia, Italy, ACM Press.

Tse, E., Greenberg, S., Shen, C. and Forlines, C. (2006) Multimodal Multiplayer Tabletop Gaming. Proceedings Third International Workshop on Pervasive Gaming Applications (PerGames'06), in conjunction with 4th Intl. Conference on Pervasive Computing, (May 7th Dublin, Ireland), 139-148.

Tse, E., Greenberg, S., Shen, C. and Forlines, C. (2007) Multimodal Multiplayer Tabletop Gaming. In ACM CIE Computers in Entertainment. June. ACM Press

The natural consequence of these arguments is that researchers are now concentrating on specialized multi-user, multimodal digital tabletop applications affording visual-spatial interaction. However, several limitations make this a challenging goal:

1. **Hardware Limitations**. Most touch-sensitive display surfaces focus on interaction with a computer mouse and thus limits interaction to a single point of contact. While some digital surfaces provide multi-touch capability (e.g., Smart Technologies DViT Board (http://www.smarttech.com), Diamond Touch [Dietz and Leigh, 2001], Smart Skin [Rekimoto, 2002], Frustrated Total Internal Reflection [Han, 2005]), this does not necessarily make it easy to detect hand postures (e.g., five fingers), especially if the technology cannot identify distinct touches from each person.
2. **Software Limitations.** It is difficult and expensive to build a truly useful collaborative multimodal application from the ground up (e.g., Quickset [Cohen *et al.*, 1997]). As a consequence, most research systems are 'toy' applications that do not afford the complete information and/or interaction possibilities expected in well-developed commercial products.

The focus of this chapter is on wrapping existing single user applications for use over a multi-user, multimodal tabletop. Just as screen/window sharing systems let distributed collaborators share views and interactions with existing familiar single user applications [Greenberg, 1991], I believe that embedding familiar single-user applications within a multi-user multimodal tabletop setting, if done suitably, can benefit co-located workers.

The remainder of this chapter develops this idea in two ways. First, I analyze and summarize the behavioural foundations motivating why collaborators should be able to use both speech and gestures atop tables. Finally, through case studies of three different systems – Google Earth, Warcraft III, and The Sims – I analyze the feasibility and limitations of leveraging such single-user applications within a multi-user, multimodal tabletop.

3.2 Behavioural Foundations

This section reviews related research and summarizes them into a set of behavioural foundations.

3.2.1 Individual Benefits

Proponents of multimodal interfaces argue that the standard windows/icons/menu/pointing interaction style does not reflect how people work with highly visual interfaces in the everyday world [Cohen, 2002]. They state that the combination of gesture and speech is more efficient and natural. I summarize below some of the many benefits gesture and speech input provides to individuals.

Deixis: speech refined by gestures. Deictic references are speech terms ('this', 'that', etc.) whose meanings are qualified by spatial gestures (e.g., pointing to a location). This was exploited in the Put-That-There multimodal system [Bolt, 1980], where individuals could interact with a large display via speech commands qualified by deictic reference, e.g., "Put that…" (points to item) "there…" (points to location). Bolt argues [Bolt, 1980], and Oviatt confirms [Oviatt, 1999] that this multimodal input provides individuals with a briefer, syntactically simpler and more fluent means of input than speech alone. Studies also show that parallel recognition of two input signals by the system yields a higher likelihood of correct interpretation than recognition based on a single input mode [Oviatt, 1999].

Complementary modes. Speech and gestures are strikingly distinct in the information each transmits, how it is used during communication, the way it interoperates with other communication modes, and how it is suited to particular interaction styles. For example, studies clearly show performance benefits when people indicate spatial objects and locations – points, paths, areas, groupings and containment – through gestures instead of speech [Oviatt, 1999 and 1997, Cohen *et al.*, 2000 and 1997]. Similarly, speech is more useful than gestures for specifying abstract actions.

Simplicity, efficiency, and errors. Empirical studies of speech/gestures *vs.* speech-only interaction by individuals performing map-based tasks showed that multimodal input resulted in more efficient use of speech (23% fewer spoken words), 35% less disfluencies (content self corrections, false starts, verbatim repetitions, spoken pauses, etc.), 36% fewer task performance errors, and 10% faster task performance [Oviatt, 1999].

Expressive gestures and hand postures. Unlike the current deictic 'pointing' style of mouse-based and pen based systems, observations of people working over maps showed that people used different hand postures as well as both hands coupled with speech in very expressive ways [Cohen *et al.*, 2002].

Natural interaction. During observations of people using highly visual surfaces such as maps, people were seen to interact with the map very heavily through both speech and gestures. The symbiosis between speech and gestures are verified in the strong user preferences stated by people performing map-based tasks, 95% preferred multimodal interaction *vs.* 5% preferred pen only. No one preferred a speech only interface [Oviatt, 1997].

Gestures and Speech are a Single System. McNeil argues that gesture and speech are closely linked in our minds and should be viewed as aspects of a single cognitive process [McNeil, 92]. Research in people's speaking patterns indicates that:

- *Gestures occur only during speech.* People almost never gesture while listening, and 90% of a speaker's gesture occur only when the speaker is actually saying something),
- *Gestures are co-expressive.* Both speech and gesture express the same or closely related meaning
- *Gestures are often synchronous.* The stroke of a gesture often overlaps with key speech utterances.

Physiological evidence also reveals that gestures and speech develop together as children and break down together in aphasia [McNeil, 1992].

3.2.2 Group Benefits

Spatial information placed atop a table typically serves as conversational prop to the group, creating a common ground that informs and coordinates their joint actions [Clark, 1996]. Expressive collaborative interactions over this information often occur as a direct result of *workspace awareness:* the up-to-the-moment understanding one person has of another person's interaction with the shared workspace [Gutwin and Greenberg, 2004]. This includes awareness of people, how they interact with the workspace, and the events happening within the workspace over time. As outlined below, many behavioural factors comprising the *mechanics of collaboration* [Pinelle *et al.*, 2003] require speech and gestures to contribute to how collaborators maintain and exploit workspace awareness over tabletops.

Alouds. These are high level spoken utterances made by the performer of an action meant for the benefit of the group but not directed to any one individual in the group [Heath and Luff, 1991]. This 'verbal shadowing' becomes the running commentary that people commonly produce alongside their actions. For example, a person may say something like "I am moving this box" for a variety of reasons:

- to make others aware of actions that may otherwise be missed;
- to forewarn others about the action they are about to take;
- to serve as an implicit request for assistance;
- to allow others to coordinate their actions with one's own;
- to reveal the course of reasoning; or,
- to contribute to a history of the decision making process.

When working over a table, alouds can help others decide when and where to direct their attention, e.g., by glancing up and looking to see what that person is doing in more detail [Gutwin and Greenberg, 2004].

Gestures as intentional communication. In observational studies of collaborative design involving a tabletop drawing surface, Tang noticed that over one third of all activities consisted of intentional gestures [1991]. These intentional gestures serve many communication roles [Pinelle *et al.*, 2003], including:

- pointing to objects and areas of interest within the workspace;
- drawing of paths and shapes to emphasiase content;
- giving directions;
- indicating sizes or areas; and,
- acting out operations.

Deixis also serves as a communication act since collaborators can disambiguate one's speech and gestural references to objects and spatial locations [Pinelle *et al.*, 2003]. An example is one person telling another person "This one" while pointing to a specific object. Deixis often makes communication more efficient since complex locations and object descriptions can be replaced in speech by a simple gesture. For example, contrast the ease of understanding a person pointing to this sentence while saying 'this sentence here' to the utterance 'the fourth sentence in the paragraph starting with the word deixis located at the bottom of page 58'.

Gestures as consequential communication. Consequential communication happens as one watches the bodies of others moving around the work surface [Segal, 1994, Pinelle *et al.*, 2003]. Many gestures are consequential *vs.* intentional communication. For example, as one person moves her hand in a grasping posture towards an object, others can consequentially infer where her hand is heading and what she likely plans to do. Gestures are also produced as part of many mechanical actions, e.g., grasping, moving, or picking up an object. This movement also serves to emphasize actions atop the workspace. If accompanied by speech, it also serves to reinforce one's understanding of what that person is doing.

Simultaneous activity. Given good proximity to the work surface, participants often work simultaneously over tables. For example, Tang observed that approximately 50-70% of people's activities around the tabletop involved simultaneous access to the space by more than one person [Tang, 1991].

Gaze awareness. People monitor the gaze of a collaborator [Heath and Luff, 1991, Ishii *et al.*, 1993, Gutwin and Greenberg, 2004]. It lets one know where others are looking and where they are directing their attention. It helps one check what others are doing. It serves as visual evidence to confirm that others are looking at the right place or are attending one's own acts. It even serves as a deictic reference by having it function as an implicit pointing act. While gaze awareness is difficult to support in distributed groupware technology [Ishii *et al.*, 1993], it happens easily and naturally in the co-located tabletop setting [Heath and Luff, 1991, Gutwin and Greenberg, 2004].

Validation and Assistance. During conversation, people provide cues to show they understood what was said. Most question and answer pairs are implicit acts of validation since the answer confirms that the question was understood by the other person [Clark, 1996]. For example, if Mary asked "what do you think about this photo?" and John responded "its good", this would validate that Mary's question was understood. There are other forms of validation provided by non-language means. For example, a nod signifies "I understand", similarly the request "pass the pepper please" could be responded by completing the task of providing pepper.

If a person does not understand or if they require assistance they can explicitly ask or break the discourse [Clark, 1996]. For example, if John asked a question and Mary did not respond in a reasonable amount of time, he would assume that she did not hear or understand the question. In requesting assistance people will often monitor what others are doing to understand their current task state to avoid interrupting another person's activity [Gutwin, 2000].

3.2.3 Implications

The above points clearly suggest the benefits of supporting multimodal gesture and speech input on a multi-user digital table. This not only is a good way to support individual work over spatially located visual artefacts, but intermixed speech and gestures comprise part of the glue that makes tabletop collaboration effective. Taken all together, gestures and speech coupled with gaze awareness support an expressive multi-person choreography of often simultaneous collaborative acts over visual information. Collaborators' intentional and consequential gesture, gaze movements and verbal alouds indicate intentions, reasoning, and actions. Participants monitor these acts to help coordinate actions and to regulate their access to the table and its artefacts. Participants' simultaneous activities promote interaction ranging from loosely coupled semi-independent tabletop activities to a tightly coordinated dance of dependant activities.

While supporting these acts are good goals for digital table design, they will clearly be compromised if we restrict a group to traditional single-user mouse and keyboard interaction. In the next section, I describe an infrastructure that lets us create a speech and gesture multimodal and multi-user wrapper around these single-user systems. As we will see in the following case studies, these afford a subset of the benefits of multimodal interaction.

3.3 Case Studies

In this section, I illustrate the behavioural foundations described earlier through three speech and gesture wrappers built atop of three commercial single user applications: Google Earth (http://earth.google.com)–a geospatial mapping application for consumers, Blizzard's Warcraft III (http://www.blizzard.com/war3)–a real time strategy game, and The Sims by Maxis (http://thesims.ea.com)–a virtual home simulator. Each case study is used to illustrate the three different aspects of the behavioural foundations:

1. **Complementary modes**. Illustrated with Google Earth, explores how speech and gesture differ in their ability to transmit and communicate information.
2. **Concurrent multimodal interaction**. Illustrated with Warcraft III, explores how speech and gestures can be used in parallel by the same person.
3. **Interleaving Actions**. Illustrated with The Sims, explores how multimodal commands can be closely interleaved across different individuals.

The following sections briefly describe their functionality and how the multimodal interface interacts with them. While the remainder of this chapter primarily focuses on two people working over these applications, many of the points raised apply equally to groups of three or four. I defer the implementation of these case studies to Chapter 5 where I describe the GSI DEMO infrastructure.

Google Earth, The Sims and Warcraft III are intended for single user interaction. By wrapping them in a multimodal, multi user digital tabletop environment, I repurpose them for collaborative use. As we will see in §3.4 however, this approach has limitations.

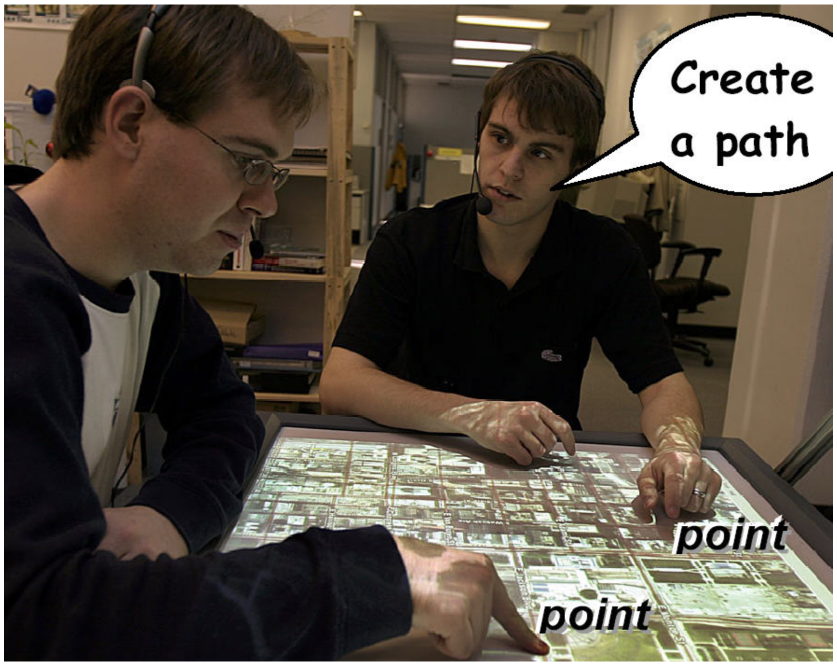


Figure 3.1 Multiple people using Google Earth on a digital table

3.3.1 Google Earth

Google Earth is a free desktop geospatial application that allows one to search, navigate, bookmark, and annotate satellite imagery of the entire planet using a keyboard and mouse. Its database contains detailed satellite imagery with layered geospatial data (e.g., roads, borders, accommodations, etc). It is highly interactive, with compelling real time feedback during panning, zooming and 'flying' actions, as well as the ability to tilt and rotate the scene and view 3D terrain or buildings. Previously visited places can be bookmarked, saved, exported and imported using the places feature. One can also measure the distance between any two points on the globe.

Complementary Modes. The behavioural foundations state that speech and gesture differ in their ability to transmit and communicate information, and in how they interact to preserve simplicity and efficiency [Oviatt, 1999, Cohen *et al.*, 1997 and 2000]. Within Google Earth, I reserve gestures primarily for spatial manipulations: panning, zooming, annotating, deixis and selection. People can pan a map using a single finger and zoom the surface in and out using two fingers analogous to how one might stretch a sheet of rubber. 'Discrete' commands that do not require direct manipulation are moved onto the speech channel (e.g., layer roads, fly to Boston, Next bookmark).

Table 3.1. The Speech/Gesture interface to Google Earth

Speech commands		**Gesture commands**	
Fly to *<place name>*	Navigates to location, eg., Boston, Paris	**One finger move / flick**	Pans map directly / continuously
Places *<place name>*	Flys to custom-created places, e.g., MERL	**One finger double tap**	Zoom in 2x at tapped location
Navigation panel	Toggles 3D Navigation controls, e.g., rotate	**Two fingers, spread apart**	Zoom in
Layer *<type>*	Toggles a layer, e.g., bars, banks	**Two fingers, spread together**	Zoom out
Undo layer	Removes last layer	**Above two actions done rapidly**	Continuous zoom out / in until release
Reorient	Returns to the default upright orientation	**One hand**	3D tilt down
Create a path *<points>* **Ok**	Creates a path that can be travelled in 3D	**Five fingers**	3D tilt up
Tour last path	Does a 3D flyover of the previously drawn path	**Bookmark**	Pin + save current location
Create a region *<points>*	Highlight via semi-transparent region	**Last bookmark**	Fly to last bookmark
Measure Distance <two points>	Measures the shortest distances between two points on the map	**Next bookmark**	Fly to previous bookmark

Table 3.1 provides a list of how I mapped Google Earth onto the multimodal speech and gesture system, while Figure 3.1 illustrates Google Earth running on a multimodal, multi user table where one person says the 'create a path" speech command and both people participate in drawing the path. Due to reasons that will be explained in §3.4.4, almost all speech and gesture actions are independent of one another and immediately invoke an action after being issued. Exceptions are 'Create a path / region' and 'measure distance', where the system waits for finger input and an 'ok' or 'cancel' utterance.

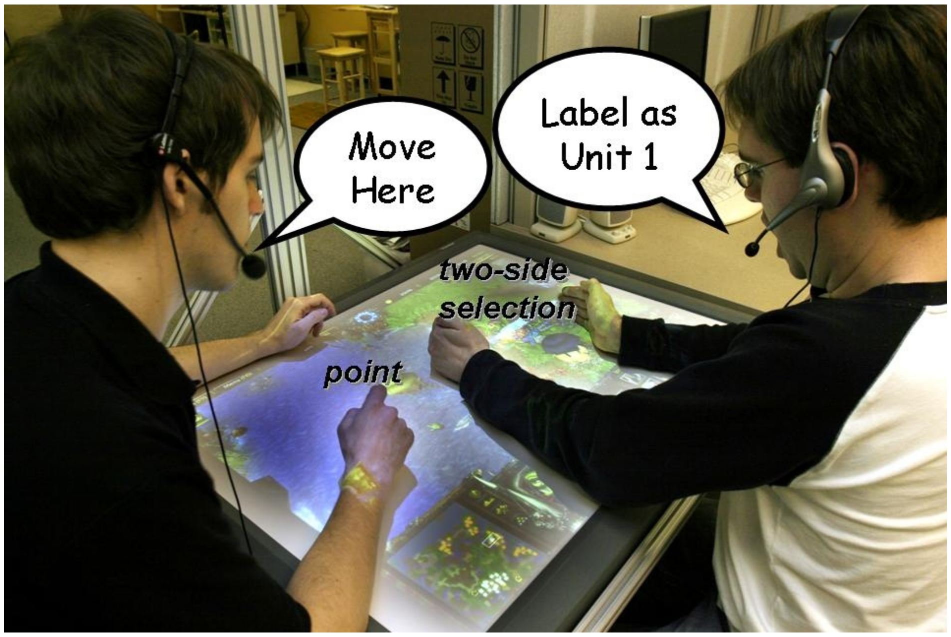


Figure 3.2 Multiple people using Blizzard's Warcraft III on a digital table

3.3.2 Warcraft III

Warcraft III is a real time strategy game. It implements a command and control scenario over a geospatial landscape. The landscape is presented in two ways: a detailed view that can be panned, and a small inset overview (Figure 3.2, lower middle). No continuous zooming features like those in Google Earth are available. Within this setting, a person can create *units* comprising semi-autonomous characters, and direct characters and units to perform a variety of actions (e.g., move, build, attack). While Google Earth is about navigating an extremely large and detailed map, Warcraft is about giving people the ability to manage, control and reposition different units over a geospatial area.

Concurrent Multimodal Interaction. Table 3.2 shows how I mapped Warcraft III onto speech and gestures, while Figure 3.2 illustrates two people using concurrent multimodal commands. The left person is pointing at a location on the digital table while saying "move here" while the right person is selecting a group of units on a table and performing the "label as unit one" speech command. Unlike Google Earth and again for reasons that will be discussed in §3.4.4, Warcraft's speech and gesture commands are performed concurrently. For example, a person may tell a unit to attack, where the object to attack can be specified before, during or even after the speech utterance.

Table 3.2. The Speech/Gesture interface to Warcraft III

Speech commands		Gesture commands	
Unit <#>	Selects a numbered unit, e.g., one, two	**One hand**	Pans map directly
Attack / attack here [point]	Selected units attack a pointed to location	**One finger**	Selects units & locations
Build *<object>* **here [point]**	Build object at current location, e.g., farm, barracks	**Two fingers**	Context –dependant move or attack
Move / move here [point]	Move to the pointed to location	**Two sides of hand**	Select multiple workers in an area
[area] Label as unit <#>	Adds a character to a unit group	**Next worker**	Navigate to the next worker
Stop	Stop the current action	**Build** *<object>* **Array**	Commands four workers to build four objects instances in the immediate vicinity

I added a number of expressive hand gestures to player's interactions of Warcraft III. The important point is that a gesture is not only recognized as input, but is easily understood as a communicative act providing explicit and consequential information of one's actions to the other players. I emphasise that the choice of gestures are not arbitrary. Rather, I examined the multimodal interactions reported in ethnographic studies of brigadier generals in real world military command and control situations [Cohen *et al.*, 2002].

To illustrate, observations revealed that multiple controllers would often use two hands to bracket a region of interest. I replicated this gesture in the tabletop wrapper. Figure 3.2 (right) and Figure 3.3 (left) show a Warcraft III player selecting six friendly units within a particular region of the screen using a two-handed selection gesture, while Figure 3.3 (right) shows a one handed panning gesture similar to how one moves a paper map on a table. This allows for these gestures to act as both commands to the computer and as communication to others around the digital table.

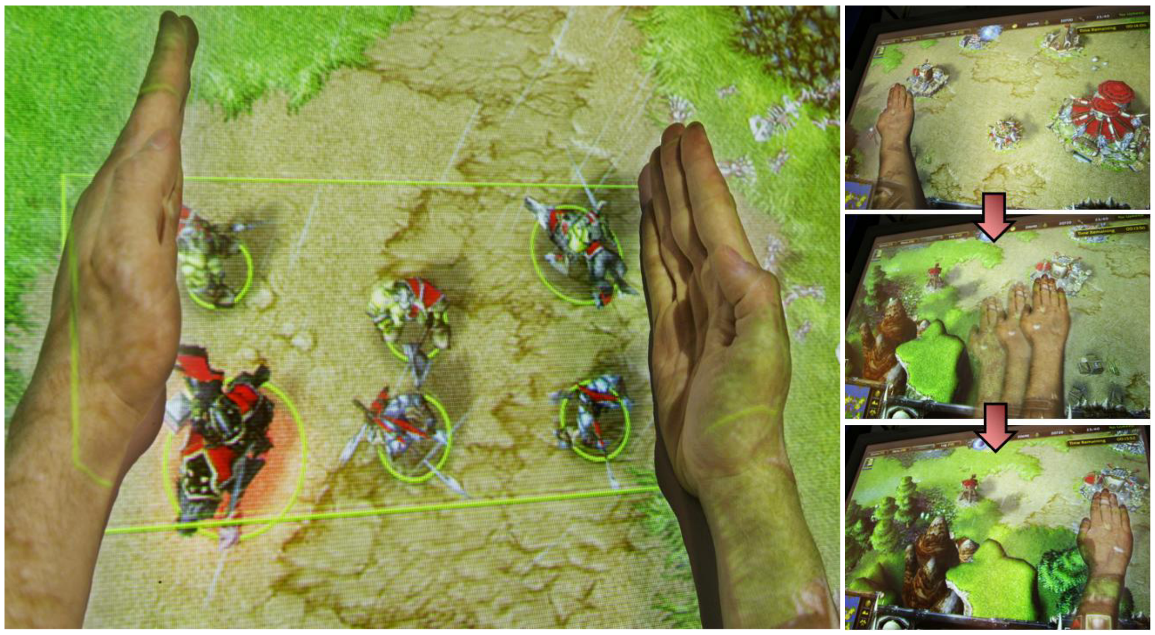


Figure 3.3 Warcraft III, 2-hand region selection gesture (left), and 1-hand panning gesture (right)

The speech and gesture commands of Warcraft III are often intertwined. For example in Warcraft III, a person may tell a unit to attack, where the object to attack can be specified before, during or even after the speech utterance. As mentioned in Section 3.2.2, speech and gestures can interact to provide an expressive language for interaction and collaboration, e.g., through deixis. Figure 3.2 gives several examples, where deictic speech acts are accompanied by one and two-finger gestures and by fist stamping; all gestures indicate locations not provided by the speech act. Further combinations are illustrated in Table 3.2. For example, a person may select a unit, and then say 'Build Barracks' while pointing to the location where it should be built. This intermixing not only makes input simple and efficient, but makes the action sequence easier for others to understand.

These multimodal commands greatly simplify the player's task of understanding the meaning of an overloaded hand posture. A user can easily distinguish different meanings for a single finger using utterances such as 'unit two, move here' and 'next worker, build a farm here' (Figure 3.4, left) since the speech command is used to qualify the gesture. Speech commands are also beneficial because they leverage the vocabulary that people have already developed in everyday communication.

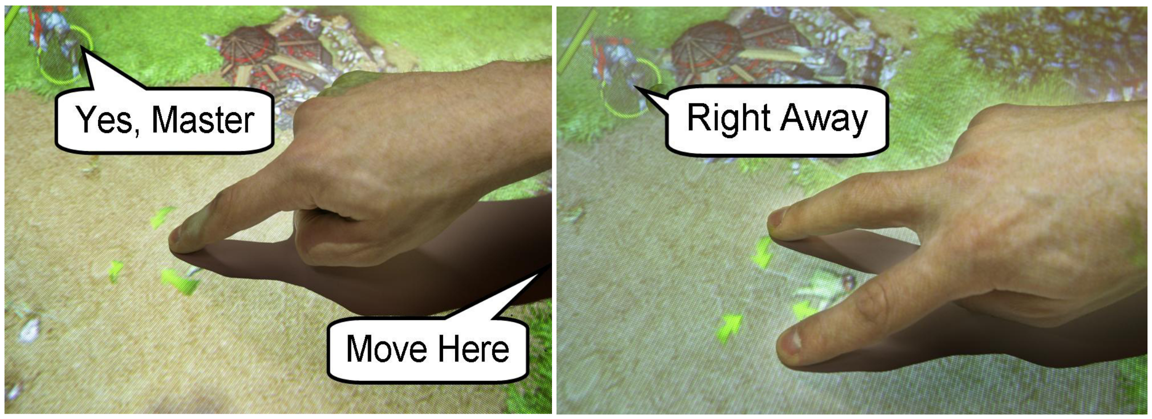


Figure 3.4 Warcraft III: 1-finger multimodal gesture (left), and 2-finger multimodal gesture (right)

For all players, game feedback re-enforces what the game understands. While feedback is usually intended for the player who did the action, it becomes feedthrough when others see and understand it. Feedback and feedthrough is done by the visuals (e.g., the arrows surrounding the pointing finger in Figure 3.4, the bounding box in Figure 3.3 left, the panning surface in Figure 3.3 right). As well, each game provides its own auditory feedback to spoken commands: saying 'Unit One Move Here' in Warcraft III results in an in-game character responding with phrases such as 'Yes, Master' or 'Right Away' if the phrase is understood (Figure 3.4).

3.3.3 The Sims

The Sims, by Electronic Arts Inc., is a real time domestic simulation game. It implements a virtual home environment where simulated characters (the Sims) live. The game visuals include a landscape presented as an isometric projection of the property and the people who live in it. Players can either control character actions (e.g., shower, play games, sleep) or modify the layout of their virtual homes (e.g., create a table). Game play is about creating a domestic environment nurturing particular lifestyles. For example, a home with entertainment facilities distributed throughout might lead to a very different lifestyle from one populated with reading materials and art. Players can quickly switch from furniture layout mode where the Sims are frozen in time to simulation mode where the Sims are active in the virtual home to see the effects of the changes they have made.

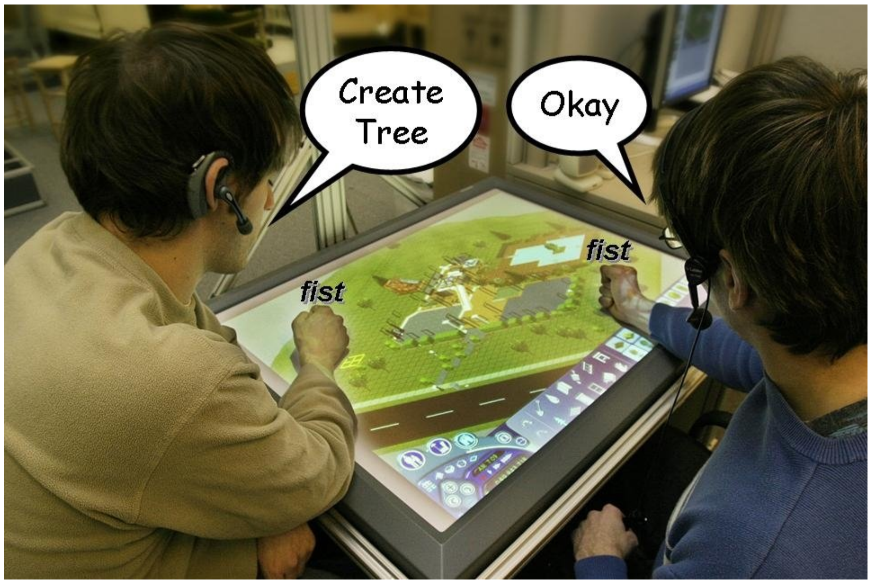


Figure 3.5 Two people using The Sims to place trees on a digital table

Table 3.3. The Speech/Gesture interface to The Sims

Speech commands		**Gesture commands**	
Rotate	Rotates the canvas clockwise 90 degrees	**One finger**	Selects/moves objects
Zoom *<In / Out>*	Zooms the canvas to one of three discrete levels	**One finger drag**	Rotates an object
<First / Second> **Floor**	Moves the current view to a particular floor	**Two fingers**	Pan the workspace
Return to Neighbourhood	Allows a saved home to be loaded	**Five Fingers**	Pick up and move an object
Create *<object>* **here [points / fists] okay**	Creates object(s) at the current location, e.g., table, pool, chair	**One Fist**	Object Stamping
Delete [point]	Removes an object at the current location	**Walls** *<Down / Up>*	Removes / Adds walls from view
<Start / Stop > **Simulation**	Enables or disables the simulation mode of The Sims	*<Slow / Fast>* **Speed**	Changes the speed of the simulation

Interleaving Actions. Table 3.3 shows how I mapped The Sims onto speech and gestures, while Figure 3.5 shows two people performing an interleaving act (create tree) across multiple people. The left person initiates the "create tree" speech command and places his fist down to start creating trees while the right person hears this and also starts placing trees. By placing his fist down the right person is confirming that he understood and agrees with what was said. Closure on their joint action is achieved through the "okay" command. This interleaving act not only splits up the multimodal command across the speech/gesture modalities and multiple people, it also splits up the decision making process of determining where to place the tree. Like Warcraft III, the speech and gesture commands in The Sims are often used concurrently. For example, a person may create a table, where the location to place the table can be specified before, during or even after the speech utterance.

The constraints and offerings of the actual commercial single player game significantly influences the appropriate gestures and speech acts that can be added to it via the wrapper. For example, continuous zooming is ideally done by gestural interaction (e.g., a narrowing of a two-handed bounding box like in Google Earth). However, since The Sims provides only three discrete levels of zoom it was appropriate to provide a meaningful aloud for zooming. Panning in The Sims is normally done with a middle mouse button similar to the scrolling in Mozilla Firefox (see §5.5.2 and §5.4.1). That is, pulling the mouse towards the user will push the virtual environment away from the user. For this reason I trained an inverted gesture action, where the position of the gesture will be inverted to the actual cursor location. This provides the illusion that the player can directly move the virtual environment despite the fact that it is not supported by the underlying application.

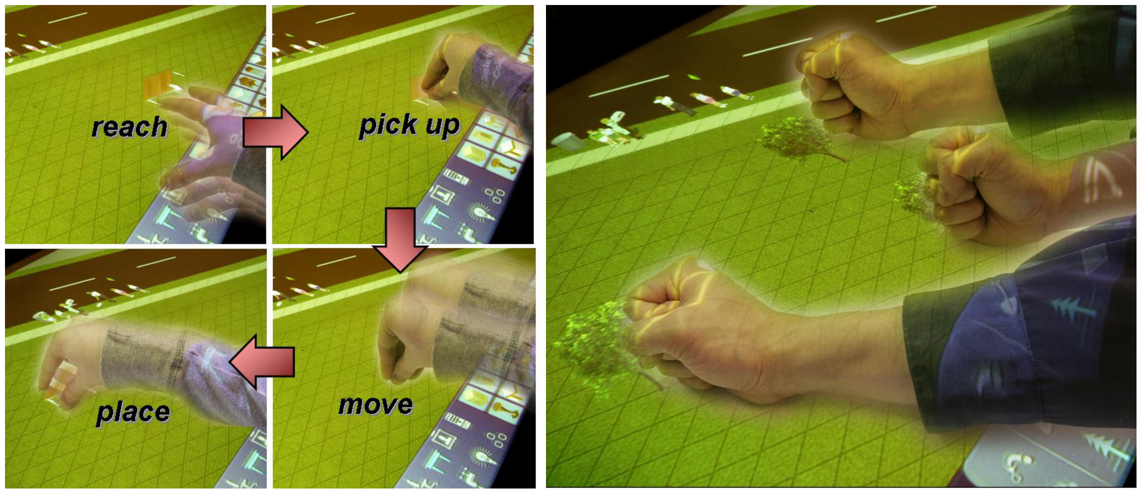


Figure 3.6 The Sims fiver finger grabbing gesture (left) and one fist stamping gesture (right)

The gesture actions in The Sims are designed to serve dual purpose as both commands to the computer and as meaningful acts of communication to other collaborators. The five finger grabbing gesture to reach, pick up, move and place virtual items on a surface simulates how people move objects in the physical world (Figure 3.6, left). A fist gesture mimics the use of a physical stamp to paste multiple object instances on the terrain (Figure 3.5 & Figure 3.6, right). Because most of these acts work over a spatial location, and the location of a gesture becomes highly meaningful to other collaborators. By overhearing alouds, by observing players' moving their hands onto the table (consequential communication), by observing players' hand postures and resulting feedback (feedthrough), participants can easily determine the modes, actions and consequences of other people's actions.

As mentioned earlier, the multimodal actions in The Sims need not be executed by a single individual. Since these public gesture and speech acts provide awareness and consequential communication commands can be closely interleaved across multiple collaborators.

3.4 Constraints of Single User Applications

From my experiences implementing, demonstrating, and observing people's reactions to multi-user multi-modal wrappers for Google Earth, Warcraft III, and The Sims, I encountered a number of limitations that influenced my wrapper design. When possible, I present solutions to mitigate these limitations, which can also guide the design of future multi-user multi-modal interactions built atop single user applications.

This section is loosely structured as follows. The first three subsections raise issues that are primarily a consequence of constraints raised by how the single user application produces *visual output*: upright orientation, full screen views, and feedthrough. The remaining subsections are a consequence of constraints raised by how the application considers *user input:* interacting speech and gestures, mapping, and turntaking.

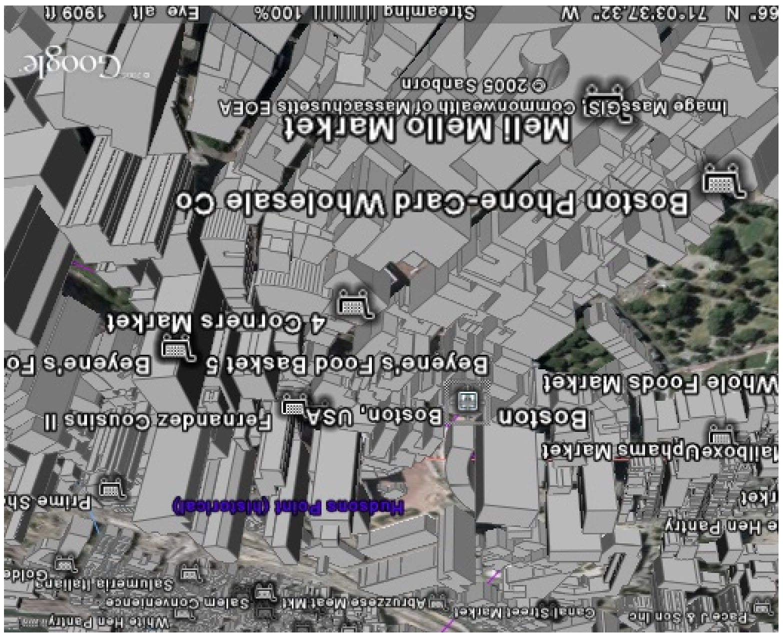


Figure 3.7 An upside down view of 3D Buildings in Google Earth

3.4.1 Upright Orientation

Most single user systems are designed for an upright display rather than a table. Thus all display items and GUI widgets are oriented in a single direction usually convenient for the person seated at the 'bottom' edge of the display, but would be upside down for the person seated across from them. As illustrated in Figure 3.7, an upside screenshot from Google Earth, problems introduced include text readability (but see [Wu *et al.*, 2006]), difficulties in comprehending incorrectly oriented 3D views, inhibiting people from claiming ownership of work areas, and preventing people from naturally adjusting orientation as part of their collaborative process [Kruger *et al.*, 2004]. Similarly, the layout of items on the surface usually favours a single orientation, which has implications for how people can see and reach distant items if they want to perform gestures over them. The isometric viewing angle in The Sims also makes it difficult to see from all viewing angles.

As with most single user applications, Warcraft III maintains a strictly upright orientation with various spatially fixed components. This gives limitations, while people can pan, they cannot rotate the landscape. Critical interface features, such as the overview map, are permanently positioned at the bottom left corner, which is inconvenient for a person seated to the right who wishes to navigate using the overview map. The Sims and Google

Earth have similar constraints: its control panel (exposed by a speech command in Google Earth) is at the very bottom, making GUI control awkward to use for anyone but the upright user. While Google Earth allows the map to be rotated, text labels atop the map are *not* rotated. In both systems, 3D perspective is oriented towards the upright user. A tilted 3D image is the norm in Warcraft III and The Sims. While Google Earth does provide controls to adjust the 3D tilt of a building on the map, the viewpoint always remains set for the upright user.

Some of these problems are not solvable as they are inherent to the single user application, although people can choose to work side by side on the bottom edge. However, speech appears to be an ideal input modality for solving problems arising from input orientation and reach, since users can sit around any side of the table to issue commands (*vs.* reach, touch or type).

3.4.2 Full Screen Views

Many applications provide a working area typically surrounded by a myriad of GUI widgets (menus, palettes, etc.). While these controls are reasonable for a single user, multiple people working on a spatial landscape expect to converse over the scene itself. Indeed, one of the main motivations for a multimodal system is to minimize these GUI elements. Fortunately, many (but not all) single user applications provide a 'full screen' view, where content fills the entire screen and GUI widgets are hidden. The trade-off is that only a few basic actions are allowed, usually through direct manipulation or keyboard shortcuts (although some applications provide hooks through accessibility APIs).

Because Warcraft III and The Sims are designed as highly interactive games, they already exploit a full screen view in which all commands are accessible through keyboard shortcuts or direct manipulation. Thus speech/gesture can be directly mapped to keyboard/mouse commands. In contrast, Google Earth contains traditional GUI menus and sidebars: When opened, 42% of the screen real estate is consumed by GUI items on a 1024x768 screen! While these elements can be hidden by toggling it into full screen mode, much of Google Earth's functionality is only accessible through these menus and sidebars. My solution uses full screen mode, in which I map multimodal commands to action macros that first expose a hidden menu or sidebar, perform the necessary action on it, and then hide the menu or sidebar (via GSI DEMO as will be described in §5.4). When this stream of interface actions is executed in a single step, the interface elements and inputs are hidden.

3.4.3 Feedback and Feedthrough

Feedback of actions is important for single user systems. *Feedthrough* (the visible consequence of another person's actions) is just as important if the group is to comprehend what another person is doing [Dix *et al.*, 1998]. Collaboration-aware groupware systems can be constructed to regulate the feedback and feedthrough so it is appropriate to the acting user and the viewing participants. Within collaboration-transparent groupware over single user systems, we can only use what is provided.

Fortunately, Google Earth, Warcraft III and The Sims are highly interactive, immediately responding to all user commands in a very visual and often compelling manner. Panning produces an immediate response, as does zooming or issuing a 'Fly to' command in Google Earth. Warcraft III visually marks all selections, re-enforcing the meaning of a gestural act. Warcraft III also gives verbal feedback. For example, if one says the 'Move here' or 'Attack here' voice command and points to a location (Table 3.2), the units will respond with a prerecorded utterance such as 'yes, master' and will then move to the specified location (Figure 3.4).

In both systems, some responses are animated over time. For example, 'Fly to, Calgary from a distant location will begin an animated flyover by first zooming out of the current location, flying towards Calgary, and zooming into the centre of the city. Similarly, panning contains some momentum in Google Earth, thus a flick gesture on the table top will send the map continually panning in the direction of the flick. In Warcraft III, if one instructs 'Unit one, build a farm here [point]', it takes time for that unit to run to that location and to build the farm. These animations provide excellent awareness to the group, for the feedthrough naturally emphasises individual actions [Gutwin and Greenberg, 1998].

Animations over time also provide others with the ability to interrupt or modify the ongoing action. For example, animated flyovers, continuous zooming or continuous panning in Google Earth can be interrupted by a collaborator at any point by touching on the table surface. Similarly a 'stop' voice command in Warcraft III can interrupt any unit's action at any time. In contrast, most productivity applications perform actions immediately leaving little or no opportunity for interruption/modification. Consider an office productivity application where most of the screen content can be changed with a single button (e.g., page down, close application, dialog boxes), these commands could seriously impede the work of others on a multi user multimodal digital table.

Feedback, even when it is missing, is also meaningful as it indicates that the system is waiting for further input. For example, if one says 'Unit one, move' to Warcraft III, the group will see unit one selected and a cross hair indicating that it is waiting for a location to move to, but nothing will actually happen until one points to the surface. This also provides others with the ability to interrupt, and even to take over the next part of the dialog.

3.4.4 Interacting Speech and Gestures

Ideally, one would like to have the system respond to individual sequential and possibly overlapping speech and gesture acts. For example, 'Put that' *<points to object>* 'there' *<points to place>* [Bolt, 1980] shows an overlapping multimodal command. This is how deixis and consequential communication works. It may even be possible to have multiple people contribute to command construction through turn taking (see §3.4.6). However, the design of the single user application imposes restrictions on how this can be accomplished.

Google Earth only allows one action to be executed at a time; no other action can be executed until that action is completed. For example if a person performs overlapping keyboard and mouse interactions only the keyboard commands will be issued. The design consequence is that I had to map most spoken and gestural actions into separate commands in Google Earth (Table 3.1). As mentioned, with the exception of the 'create a path/region' and 'measure distance' command, gestures and speech do not interact directly. Some gesture and speech commands move or zoom to a location. Other speech commands operate in the context of the current location, usually the center of the screen. For example, 'bookmark' only acts on the screen center; while a person can position the map so the location is at its center, they cannot say 'Bookmark' and point to a location off to the side.

In contrast, Warcraft III and The Sims are designed to be used with the keyboard and mouse in tandem, i.e., it can react to overlapping keyboard and mouse commands. This makes it possible to use intermixed speech and deixis for directing units. My mapping uses speech in place of keyboard commands, and gesture in place of mouse commands, e.g., saying 'Unit 1, move here' while pointing to location in Warcraft III. Object creation in The Sims is performed through a series of mouse commands, GSI DEMO queues all mouse commands until the multimodal command has been completed. This way overlapping multimodal commands are executed sequentially to the single user application.

By understanding the sometimes subtle input constraints of the single user application, a wrapper designer can decide if and where intermixing of speech and gestures via mapping is suitable.

3.4.5 Human Understandable Mappings

Mapping of Gestures. Many gesture based systems rely on abstract gestures to invoke (i.e., start mode change) commands. For example, a two fingered gesture invokes an 'Annotate' mode in Wu *et al.*'s example application [2006]. Yet the behavioural foundations state that people working over a table should be able to easily understand other people's expressive gestural acts and hand postures as both consequential communication and as communicative acts. This strongly suggests that the vocabulary of postures and dynamics must reflect people's natural gesture acts as much as possible (a point also advocated in [Wu *et al.*, 2003 and 2006]).

Because I reserve gestures for spatial manipulations, very little system learning is needed: panning by dragging one's finger or hand across the surface is easily understood by others, as is the surface stretching metaphor used in spreading apart or narrowing two fingers to activate discrete or continuous zooming in Google Earth. Pointing to indicate deictic references, and using the sides of two hands to select a group of objects in Warcraft III is also well understood [Cohen *et al.*, 1997 and 2000, Oviatt, 1999]. Because most of these movements work over a location, gaze awareness becomes highly meaningful. However, the table's input constraints can restrict what one would like to do. For example, an upwards hand tilt movement would be a natural way to tilt the 3D map of Google Earth, but this posture is not recognized by the DiamondTouch table. Instead, I resort to a more abstract one hand / five finger gesture set to tilt the map up and down (Table 3.1).

Mapping of Speech. A common approach to wrapping speech atop single user systems is to do a 1:1 mapping of speech onto system-provided command primitives. This is inadequate for a multi-user setting. A person should be able to rapidly issue semantically meaningful commands to the table, and should easily understand the meaning of other people's spoken commands within the context of the visual landscape and their gestural acts. In other words, speech is intended not only for the control of the system, but also for the benefits of one's collaborators. If speech were too low level, the other participants would have to consciously reconstruct the intention of the user. The implication is that speech commands must be constructed so that they become meaningful 'alouds'.

Within Google Earth, I simplified many commands by collapsing a long sequential interaction flow into a macro invoked by a single well formed utterance (Table 3.1). For example, with a keyboard and mouse, flying to Boston while in full screen mode requires the user to: 1) use the tool menu to open a search sidebar, 2) click on the search textbox, 3) use the keyboard to type in 'Boston, MA' followed by the return key, and 4) use the tool menu to close the search sidebar. Instead, a person simply speaks the easily understood two-part utterance 'Fly to' 'Boston'. I also created 'new' commands that make sense within a multimodal multi-user setting but are not provided by the base system. For example, I added the ability for anyone to undo layer operations (which adds geospatial information to the map) by creating an 'Undo Layer' command (Table 3.1). Under the covers, the mapping module remembers the last layer invoked and toggles the correct checkbox in the GUI to turn it off.

Intermixing of Speech and Gesture. I explained previously that a strength of multimodal interaction is that speech and gestures can interact to provide an expressive language for interaction and collaboration. Because of its ability to execute overlapping commands, Warcraft III provides a good example of how speech and gesture can be mapped to interact over a single user application. The Warcraft III speech vocabulary was constructed as easily understood phrases: nouns such as 'unit one', verbs such as 'move', and action phrases such as 'build farm' (Table 3.2). These speech phrases are usually combined with gestures describing locations and selections to complete the action sequence. For example, a person may select a unit, and then say 'Build Barracks' while pointing to the location where it should be built. This intermixing not only makes input simple and efficient, but makes the action sequence easier for others to understand.

3.4.6 Observations of Turn taking

Single user applications expect only a single stream of input coming from a single person. In a multi-user setting, these applications cannot distinguish which commands come from each person, nor can they make sense of overlapping commands and/or command fragments that arise from simultaneous user activities.

In shared window systems, confusion arising from simultaneous user input across workstations is often regulated through a *turn taking* wrapper interposed between the multiple workstation input streams and the single user application [Greenberg *et al.*, 1990 and 1991]. Akin to a switch, this wrapper regulates *user pre-emption* so that only one workstation's input stream is selected and sent to the underlying application. The wrapper could embody various turn taking protocols, e.g., explicit release (a person explicitly gives up the turn), pre-emptive (a new person can grab the turn), pause detection (explicit release when the system detects a pause in the current turn-holder's activity), queue or round-robin (people can 'line up' for their turns), central moderator (a chairperson assigns turns), and free floor (anyone can input at any time, but the group is expected to regulate their turns using social protocol) [Greenberg, 1991].

In the distributed setting of shared window systems, technical enforcement of turn taking is often touted since interpersonal awareness is inadequate to effectively use social mediation. The three case studies reveal far richer opportunities for social regulation of turn-taking in tabletop multimodal environments.

Ownership through Awareness. Unlike distant-separated users of shared window systems, co-located tabletop users are more aware of moment by moment actions of others and thus are better able to use social protocol to mediate their interactions. Alouds arising from speaking into the headset lets others know that one had just issued a command so they could reconstruct its purpose, thus people are less likely to verbally overlap one another, or to unintentionally issue a conflicting command. Through consequential communication, people see that one is initiating, continuing or completing a gestural act; this strongly suggests one's momentary 'ownership' of the table and thus regulates how people time appropriate opportunities for taking over. The real time visual feedback and feedthrough provided by both Google Earth and Warcraft emphasises who is in control, what is happening, when the consequences of their act is completed, and when it is appropriate to intercede.

Interruptions. Awareness not only lets people know who is in control, but also provides excellent opportunities for interruptions. That is, a person may judge moments where they can stop, take over and/or fine-tune another person's actions. Eye gaze and consequential communication helps people mutually understand when this is about to happen, enabling cooperation *vs.* conflict. For example, animations initiated by user actions (e.g., unit movement in Warcraft or the animated flyovers in Google Earth) can be interrupted by a spoken command ('Stop') or a gestural command (touching the surface).

Assistance. Awareness also provides opportunities for people to offer assistance. Indeed, the interruptions mentioned above are likely a form of assistance, i.e., to repair or correct an action initiated by another person [Clark, 1996]. Assistance also occurs when multiple people interleave their speech and gestures to compose a single command. For example, I previously mentioned in §3.3.3 how multi modal commands in The Sims are actually phrases that are chained together to compose a full command. In Warcraft III as one person starts a command ('unit one', 'move') another can continue by pointing to the place where it should move to. Similarly, the 'create a path' and 'create a region' spoken commands in Google Earth expect a series of points: all members of the group can contribute these points through touch gestures.

The Mode problem. In spite of the above, people can only work within the current mode of the single user application. While one can take over (through turn taking) actions within a mode, two people cannot work in different modes at the same time. For example, in Warcraft III it is not possible for multiple people to control different units simultaneously. Also, in The Sims it is not possible to pan and create objects at the same time.

While these case studies suggest that social regulation of turn taking suffices for two people working over a multi modal, multi user tabletop (since the group has enough information to regulate themselves), there could be situations where technical mediation is desired. Examples could include larger groups (to avoid accidental command overlap and interruptions), participants with different roles, or conflict situations. The turn taking module provided by GSI DEMO (as will be described in §5.4.2) made it easy to incorporate turn taking to the Application Mappings of these case studies. This module knows which user is trying to interact with the system by touch or speech, and can detect when multiple people are contending for the turn. Decision logic or coordination policies [Greenberg, 1991, Ringel-Morris *et al.*, 2004] can then decide which input to forward to the application, and which to ignore (or queue for later). The logic could enforce turn taking policies at different levels of granularity.

1. *Floor control* dictates turns at a person level, i.e., a person is in control of all interaction until that turn is relinquished to someone else.

2. *Input control* is when one input modality has priority over another modality, e.g., gesture takes priority over speech commands.

3. *Mode control* enforces turn taking at a finer granularity. If the system detects that a person has issued a command that enters a mode, it blocks or queues all other input until the command is complete and the mode is exited. For example, if a person opens the navigation panel or begins a tour flyover in Google Earth, all input is blocked until the flyover is completed.

4. *Command control* considers turn taking within command composition. If the system detects that a person has issued a phrase initiating a command, it may restrict completion of that command to that person. For example, if a person says "create hot tub" in The Sims, the system may temporarily block others from issuing commands until the hot tub has been placed on the digital surface. Alternately, other people may be allowed to interleave a subset of command phrases to that character, e.g., while they can gesture to enter points via Google Earth's "Create a Path" command, only the initiator of the speech command can complete that command with the spoken "Okay".

3.5 Conclusions

In this chapter I explained why speech and gestural interaction is useful in a co-located collaborative setting. By surveying the literature on groupware and multimodal interactions, I presented key behavioural affordances that motivate and inform the use of multimodal, multi user table top interaction. These behavioural affordances were applied in practice to multi user multimodal wrappers for three existing single user systems (Google Earth, Warcraft III, and The Sims). From my experience, I derived a detailed but generalized analysis of issues and workarounds, which in turn provides guidance to future developers of this class of systems.

This work represents an important first step bringing multimodal multi-user interaction to a table display. By leveraging the power of popular single user applications, I bring a visual and interactive richness to the digital table that could not be achieved by a simple research prototype. Consequently, demonstrations of these systems to the creators of Google Earth, real world users of geospatial systems including New York Police Department officers in the Real Time Crime Center, and Department of Defence members have evoked overwhelming positive and enthusiastic comments, e.g., "How could it be any more intuitive?"

However, the success of using existing single user applications over a multi user multimodal digital table is limited to simple mappings of a subset of the application's functionality. Speech recognition performance degrades with very large speech vocabularies, and it is difficult for people to memorize more than a few simple commands. For example, speech recognition performance would degrade if I mapped every possible location in Google Earth using the "fly to <*place*>" speech command, I have to limit the number of possible fly to locations in the actual wrapper. Games optimized for rapid keyboard and mouse input such as Warcraft III are less efficient on a multimodal digital table as every command needs to be first processed by speech and gesture recognizers before execution. However, I stress that the purpose of these multimodal wrappers is not to improve game performance but rather to make collaborative interactions more publicly perceptible to others.

The next chapter examines the claims made in the behavioural foundations described in this chapter through an observational study. I examine how people use the multi user multimodal wrappers discussed in this chapter for collaborative tasks and pay particular attention to see if people use multimodal commands as both commands to the computer and as communication to others.

Chapter 4. How Pairs Use a Multimodal Digital Table

***Objective Three**. I will observe how pairs use collaboration-transparent speech and gesture wrappers over existing applications on a digital table.*

The previous chapter explored how people could interact over collaboration-transparent single user applications (e.g., Blizzard's Warcraft III, Maxis's The Sims, and Google Earth) displayed on a digital table that recognized both speech and expressive hand gestures. I listed a number of behavioural foundations motivating this multi-user, multimodal interaction. In particular, I hypothesised that one person's speech and hand gestures used to ***command*** the application also produced ***consequential communication*** that others could leverage as cues for validation and assistance. While previous ethnographic studies and empirical investigations indicated that consequential communication occurs regularly in real world situations, e.g., where people interact over physical artefacts such as paper maps [Cohen *et al.*, 2002], we do not know if these behavioural benefits accrue to speech and gesture commands directed to a digital system. I performed an observational study investigating how people used two of the multi-user speech and gesture wrappers described in Chapter 3. As we will see, my analysis verifies and adds detail to the role that speech and gesture commands play as consequential communication. My results show that pairs used multimodal commands for both communication and control (as illustrated by the validating command in Figure 4.1).[2]

[2] Portions of this Chapter are published as:

Tse, E., Shen, C., Greenberg, S. and Forlines, C. (2007) How Pairs Interact Over a Multimodal Digital Table. Proceedings of ACM CHI Conference on Human Factors in Computing Systems (May 1-3, San Jose, CA), ACM Press, 215-218.

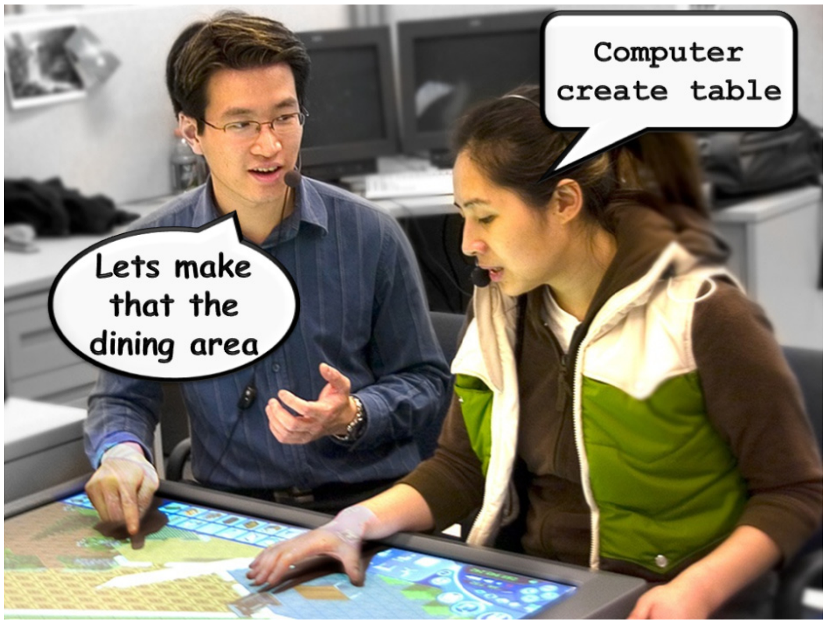


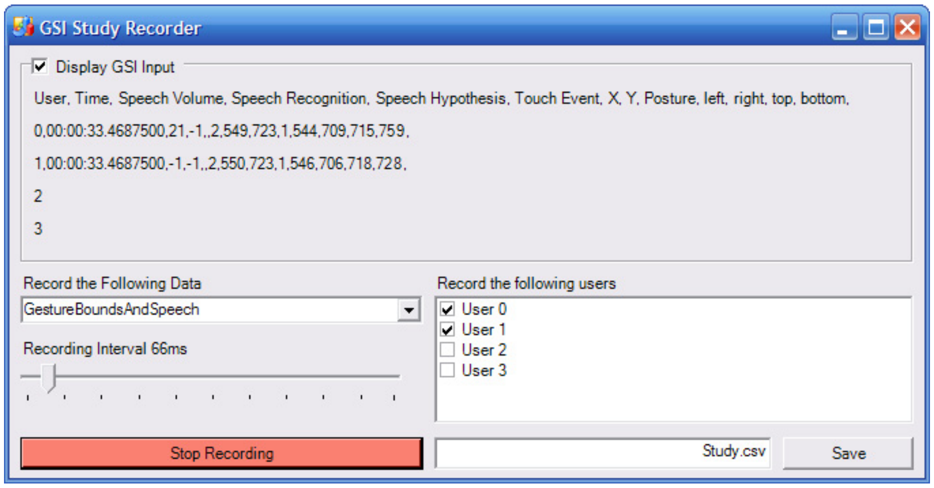
Figure 4.1 An illustration of how multimodal commands were also used for validating the decision to make a room a dining area.

4.1 Observational Study Design

I observed 6 computer-proficient participants (3 pairs): 5 males and 1 female, ages 21-30 years. Pairs were seated side by side along the front edge of the digital table displaying an 'upright' single user application (Figure 4.3, top). Participants interacted with the application using speech via noise cancelling headsets and gestures via a DT107 MERL DiamondTouch table. Speech and gesture inputs were mapped to GUI commands using GSI DEMO as described in Chapter 5. The speech recognition engine distinguished speech commands from conversational speech by listening for a 'Computer' prefix (e.g. "*Computer*, create phone"). Participants used gestures as commands by directly touching the table surface. Feedback of successful speech and gesture recognition was indicated by the application's visual response and by an audio tone for speech commands. Spoken commands were designed to be easily understood by both the computer and other collaborators (e.g., "fly to [city]"). A printed list of recognizable speech and gesture commands was posted in front of participants, and pairs were encouraged to practice speech and gesture input prior to each trial. Tasks consisted of two scenarios, described below.

Travel Planning. Pairs used Google Earth to plan a European student's three day, all expenses paid, trip to Boston, New York and Chicago. Typical speech commands were "fly to [city]" and "layer [name e.g., roads]", while gestures included using one finger to pan or annotate, two fingers to zoom the camera in and out, and five fingers to tilt the camera. A complete list of commands is listed in Table 3.1. Pairs had to select four or five key places to visit in each city by using the "scratch pad" speech command, circling the area of interest and numbering the attractions in the order they would be visited.

Home Layout. Pairs used The Sims by Maxis to lay out furniture in a bedroom, living room, kitchen, and washroom of a newly purchased two story home for a four person family. Typical speech commands were "create [object]", "[first/second] floor", "walls [down/up]", while gesture commands included two finger pan, five finger object pick up, and one fist object stamping. A complete list of commands is listed in Table 3.3.



```
User, Time, Speech Volume, Speech Recognition, Speech Hypothesis, Touch Event, X, Y, Posture, left, right, top, bottom,
0,00:00:33.4687500,21,-1,,2,549,723,1,544,709,715,759,
1,00:00:33.4687500,-1,-1,,2,550,723,1,546,706,718,728,
2
3
```

Record the Following Data: GestureBoundsAndSpeech

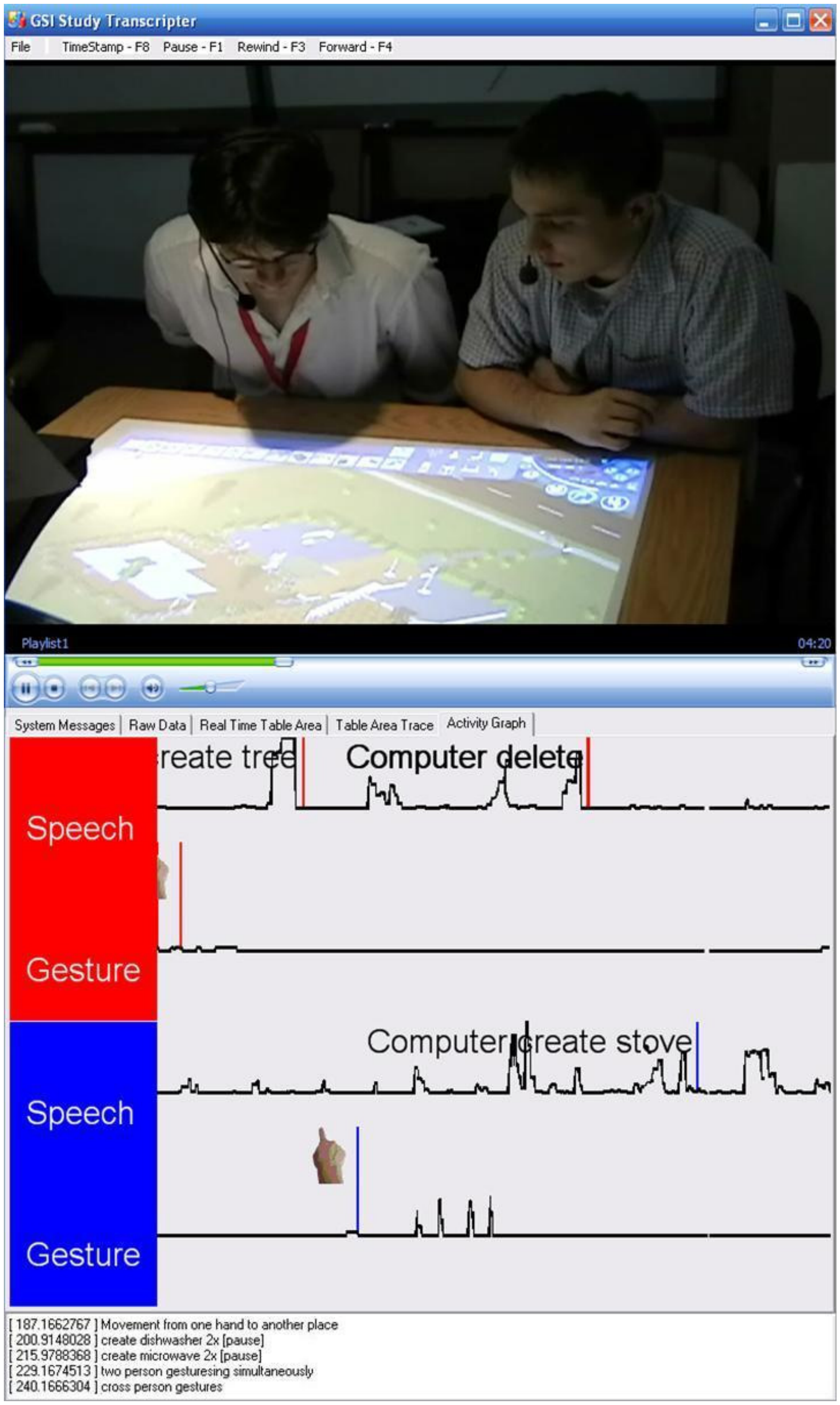
Recording Interval 66ms

Record the following users: ✓ User 0, ✓ User 1, User 2, User 3

Figure 4.2. The GSI Study Recorder User Interface

4.1.1 Video and Data Collection

While video transcription Systems such as STAMP [Clow and Oviatt, 1998] allow recorded multimodal interactions to be synchronized with videos of participants using the system for a single person, I needed a way to capture the interactions of *multiple* people interacting with applications built atop of GSI DEMO (as will be described in Chapter 5). To do this I created the GSI Study Recorder (Figure 4.2) as a client application that could be run in conjunction with other applications built using GSI DEMO.



```
[ 187.1662767 ] Movement from one hand to another place
[ 200.9148028 ] create dishwasher 2x [pause]
[ 215.9788368 ] create microwave 2x [pause]
[ 229.1674513 ] two person gesturesing simultaneously
[ 240.1666304 ] cross person gestures
```

Figure 4.3 The Study Transcription Application

The GSI Study Recorder would record the speech and gesture recognition information along with a time stamp that could be synchronized with a separately recorded video. The experimenter would press a button (Figure 4.2, bottom left) on the recorder to begin recording the actions for the current session, when pressed the button colour would change from gray to red. This visual cue could be used to synchronize the video with the data but required that the video also record the tabletop display (as was the case in Figure 4.3, top). After the session was completed the experimenter would stop the recording and save the data to a file.

After all the experimental data was recorded, the recorded data and video could be later used for analysis using the GSI Study Transcripter tool. While it is technically possible to visualize the data in real time during the experiment, I needed to perform several passes on both the data and the video for my open coding (described in §4.2). Figure 4.3 shows a screen snapshot of the logging tool. The top shows the video which can be played at a normal speech 2x faster or 2x slower; being able to slow the rate of video playback is useful for transcribing the speech actions of multiple people. Hotkeys are provided to rewind or advance the video by five seconds. This video is synchronized with a visualization of both participants' speech and gesture actions and how they were recognized by the system (an activity graph in Figure 4.3). The bottom pane includes manual transcription notes. A hotkey can be used to include the current timestamp in the transcription.

For example, Figure 4.3 (middle) is a sequence in time where the left user (upper middle) said "computer create tree" after which the right user specified the location of the tree (lower middle) with a single finger. The coloured vertical lines on each timeline represent when a speech command has been recognized (the corresponding keyboard and mouse macro is played almost immediately afterwards) or when a gesture command is released (the corresponding mouse command occurs continuously throughout the gesture). This visualization made it easier to observe interleaving (cross person) actions over the digital table as it could alert the reviewer to interleaving acts that might otherwise be missed in manual video transcription. Also, the recognized speech and gesture actions would appear in this visualization, making it easy to determine if the recognized command matched what people said in the video.

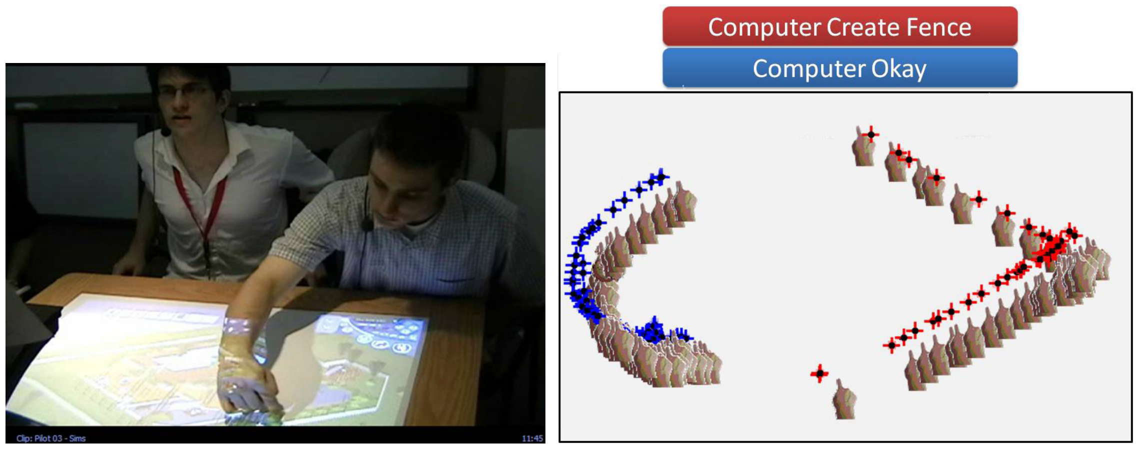


Figure 4.4 Visualization of two people drawing a fence around a virtual home in The Sims

Other views accessed by the tabs at the top of the GSI Transcripter Data Visualization (Figure 4.3, middle) allow further data exploration. The Table Area Trace view allows the experimenter to replay recorded gestures over a 2D bird's-eye-view of the table. Figure 4.4 (right) shows a table area trace of two people drawing a fence around a virtual home in The Sims, the left person said "Computer Create Fence" and created half the fence while the other person completed the fence on the other side and said "okay". This visualization also showed the gesture and speech actions recognized by the computer. Figure 4.4 (right) shows that both participants used a single finger for drawing the fence around the table, the corresponding speech recognition actions are shown above.

Reviewers could also view aggregated statistics of the experimental session (e.g., the amount of simultaneous activity, multimodal activity, interleaving speech and gesture activity), with the Raw Data view (Figure 4.3, middle) these statistics were formatted so that they could be used in existing spreadsheet applications for analysis and graphing. The raw data is also available to programmers via the recorded session, so that one can create their own visualizations or statistical analyses.

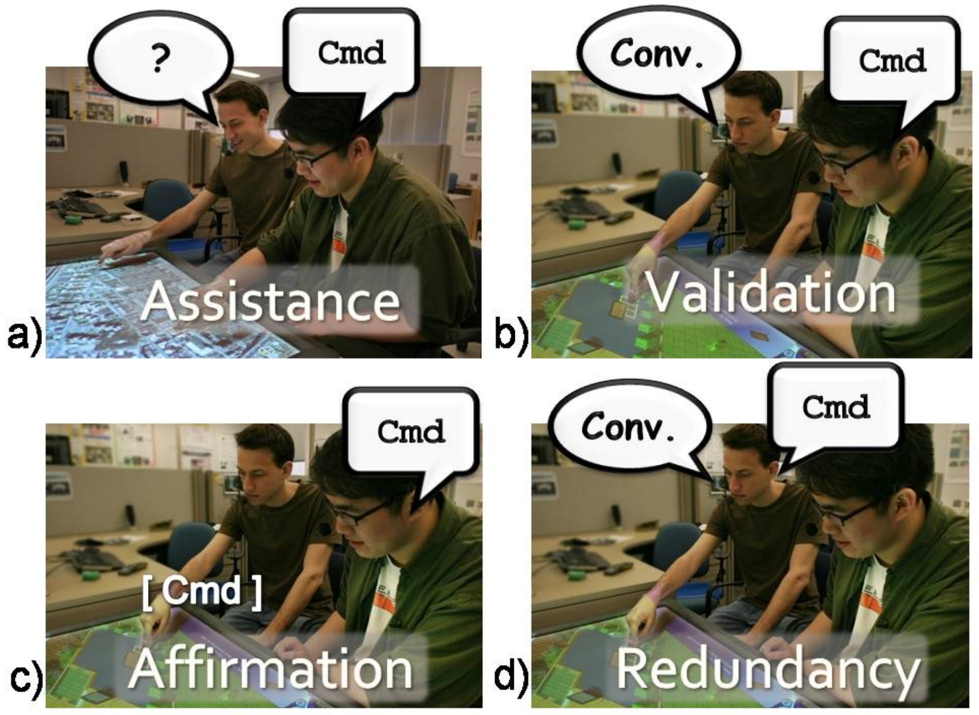


Figure 4.5 How multimodal commands were used for implicit communication

4.2 Commands as Implicit Communication

4.2.1 Method

Using the GSI Study Transcripter application, I recorded and then transcribed a total of 476 minutes of speech and gesture actions from each participant, as recognized by the system at 15 events per second.

Open Coding. I analyzed the transcriptions using an open coding method [Strauss and Corbin, 1998] to draw out similarities and differences in how people used multimodal commands. That is, for each observation I assigned it a code that stylized it, and used that code to mark any recurrence of it. Observations that did not fit were given a new code. For example, when going through the videos of people using multimodal commands, I noticed that some people would use commands as a direct answer to a question. I created a label [AS] to represent this type of information. Each time I came across another command that could be characterized as 'assistance', I flagged the data with the same code, [AS]. At times this process was iterative: I would systematically analyze data from several participants, uncover new categories, and then return to previously analyzed data for further analysis using the new codes / categories. Codes generated in the analysis are found in Section 4.2.2. This analysis method is widely used and accepted in the social sciences, thus the remainder of this chapter will focus on results instead of low level details of the raw data and its analysis.

4.2.2 Coding Categories

Over all pairs, 416 commands were coded: 164 speech, 194 gesture and 58 multimodal commands (i.e., where speech and gesture together form the command). Using the open coding method described earlier, I discovered four mutually exclusive ways in which pairs used speech and gesture commands (as illustrated in Figure 4.5, commands are illustrated with a square speech bubble and conversation is illustrated with a round speech bubble):

Assistance. As illustrated in Figure 4.5a, people invoke commands as actions that directly respond to other people's explicit or implicit requests for help. For example, the person seated on the right (R) side of the video was using Google Earth to plan a tourist's vacation in Boston. He discovers an interesting attraction and says:

R: There's… [zooms in] Fenway Park. Okay, how do you…?
L: Computer scratch pad (successful recognition)

The person seated on the left (L) side answered the question by using a speech command instead of explaining through conversation. Thus the speech command was used for assistance.

Validation. A person's use of a command validates joint understanding and agreement reached in prior conversation (Figure 4.5b). For example, two people are trying to lay out furniture in a virtual home by first establishing what each room will be. The left person says:

L: And here [points], maybe a kind of small living room?
R: Computer create couch (successful recognition)

The right person agreed with the decision made by the left person and established closure on this joint action by using the "Computer create couch" speech command. The right person is agreeing with or validating what the L person had said through conversation.

Affirmation. A person's command triggers an explicit follow-up agreement about the action or an implicit agreement when both continue with the task at hand (Figure 4.5c). For example, in The Sims the right person might decide to create a couch in the living room by saying:

R: Computer create couch (successful recognition)

L: Yeah [single finger placement in living room] good.

In this example the left person hears that the create couch command and agrees with the placement of a couch in the living room. To affirm her agreement with the create couch command she uses a single finger to decide the location for the couch in the living room. Thus, the decision to create a couch and position in a particular area in the living room is split among both participants.

Redundancy. A person explicitly mentions the action both in conversation and as a command (Figure 4.5d), i.e., saying the command is redundant. For example, in The Sims one person might be placing furniture (e.g., tables, refrigerator, sink) while engaging in self talk:

L: Yeah, let's go, kitchen is basically... oh trash can, computer create trash can (successful recognition)

Here the left person is almost ready to say that all the furniture needed in the kitchen is placed, but then he realizes that they are missing a trash can. He says this out loud and then says it as a command. Thus the speech command does not add to the communication of the group, rather it serves mainly as a command to the computer.

Assistance, validation and affirmation are all examples of commands that are positively included as conversational elements. Redundancy is an indication that a person viewed the action as distinct communication and command elements.

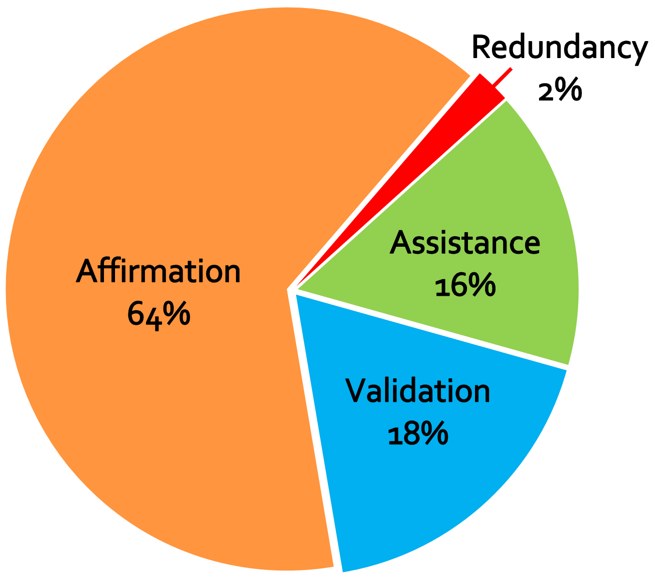


Figure 4.6 Piechart of implicit communication found in the use of speech and gesture commands

4.2.3 Frequency of Coding Occurrences

I did not formally verify the number of coding occurrences with other researchers in this exploratory study. Instead I presented the video and data to two other collaborators to review and informally discussed the multimodal actions observed to ensure that my coding categories made sense to them in light of the data they observed.

Figure 4.6 shows the average breakdown of the 416 coded speech and gesture command used across both tasks (Travel Planning and Furniture Layout). I coded 264 (64%) as affirmation, 73 (18%) as validation, 68 (16%) as assistance, 11 (2%) as redundancy. Affirmation was coded frequently because it was typical to see groups break into long sequences of speech and gesture commands. In Figure 4.7 the 58 multimodal commands are split into their speech and gesture components. Gestures were most often coded as affirmation because they typically followed each other without any need for explicit conversation or gesturing above the table.

When these numbers are considered by task (Figure 4.7 left vs. right), the Travel Planning Google Earth task had slightly higher validation and assistance rates than the Home Layout Sims task. I believe this is because many Google Earth commands performed global actions that would affect the entire work area, and for this reason, participants would converse with their partners before issuing the command.

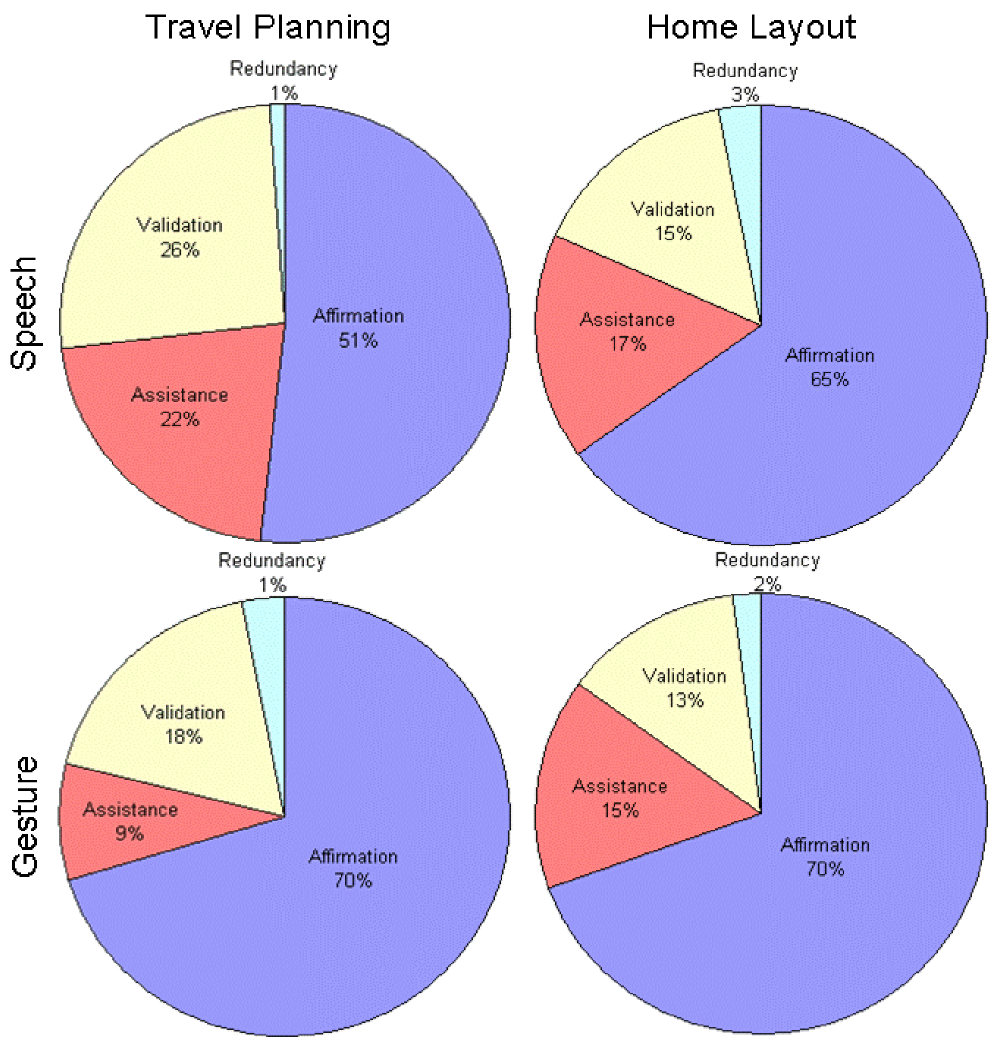


Figure 4.7 Speech / Gestures as Communicative Categories

4.2.4 Discussion

The coding results clearly show that speech and gesture commands directed to the system also served double duty as communication to other collaborators. Ninety-eight percent of our 416 observations were coded as assistance, validation, or affirmation. Only 2% - the clarification and redundancy categories – indicated commands that were not included well within the conversational context. Our own subjective appraisal of pair interactions confirms what these numbers suggest: people integrated speech and gesture commands into their joint conversations and actions.

To explain these results, Clark [1996] describes how speech acts can be broken up into two tracks: track one describes the business of the conversation and, track two describes the efforts made to improve communication. With commands, track one becomes the act of issuing a command to the computer, while track two serves a communication role to other collaborators. I deliberately crafted speech commands so they were both machine and human recognizable (e.g., fly to Boston *vs.* reposition 135436). Our results suggest that pairs' used speech commands as dual purpose speech acts that fit into both tracks.

Similarly, consequential communication happens when one monitors the bodies of other's moving around the work surface [Pinelle *et al.*, 2003, Segal, 1994]. For example, as one person moves her hand in a grasping posture towards an object, others can infer where her hand is heading and what she likely plans to do. In our system, gesture commands are designed so that they provide consequential communication to others when used. For example, using five fingers to pick up a digital couch also produces awareness to collaborators around the table.

4.3 Simultaneous Activity and Interleaving Acts

We now consider how people interact in this multimodal tabletop setting. I was particularly interested in whether the single user nature of the underlying application (i.e., where multi-user input is multiplexed into a single input stream) forced a situation in which people predominantly worked sequentially (e.g., by gross turn taking), or whether they were able to converse and interact simultaneously over this surface.

4.3.1 Simultaneous Conversation and Command Activity

First, I used the study logger to mark each person's gesture and speech actions as either on or off: speech that is used for conversation or a command is on when it is above a volume threshold, while a gesture is on whenever the logger detects a finger or hand posture placed on the table (gestures above the table are not recorded). Thus for any instant in time I can determine if an overlapping speech and gesture act is occurring. I then examined those times when at least one person was speaking and/or gesturing (53% of the time). For about 14% of this 53%, I found that the other person was also speaking and/or gesturing at the same time. i.e., they were interacting concurrently with each other either through conversation or by issuing a command. This number actually underestimates simultaneous activity, as it only includes those gestures which are direct touches to the table. In actual practice, many gestures occurred immediately above and around the table, as well as nodding and many other forms of body language. I observed (both during the experiment and from a review of the video recordings) that participants were highly engaged in each other's task and actions; it was rare to find a participant idling. They were involved both in how they attended to each other, and in the interleaving of their speech and gestures when talking about what they were doing. This supports other people's findings of simultaneous interaction over tables [Scott *et al.*, 2004, Tang *et al.*, 2006].

4.3.2 Interleaving Commands

Next, I examined how people worked together during those episodes in where at least one person was directing speech and gesture commands to the application. Here, I analyzed the video transcriptions, again using an open coding method (described in §4.2.1), to look for different styles of interleaving commands. My analysis revealed that even though the underlying application could not recognize simultaneous activity, people managed to cooperate through *interleaving commands:* a graceful mixing of people's speech and gesture actions in the construction of commands. I saw four different interleaving command interactions that can be described along the dimensions of coupling [Tang *et al.*, 2006b] and the input modality used.

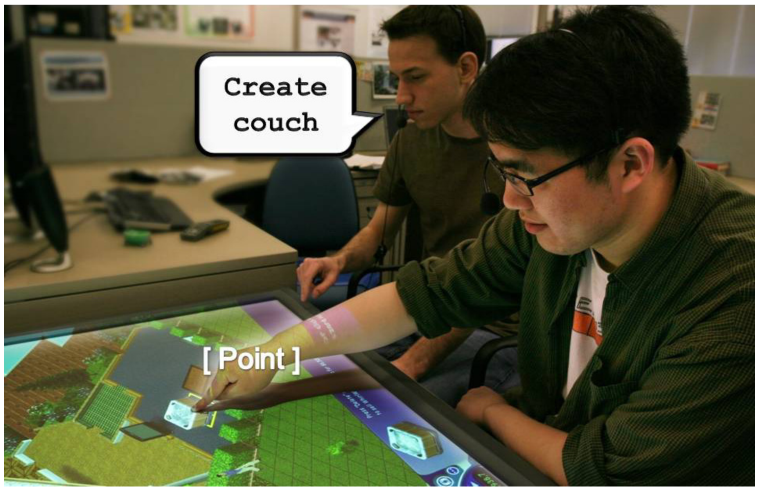


Figure 4.8 A tightly coupled, inter-modal interleaving hot tub creation act

Tightly Coupled, Inter-Modal. This category occurs when one person issues the speech component of a command and the other issues the gesture component. For example, the following interaction separates one's decision of creating a chair from the specification of the location for it (as illustrated in Figure 4.8).

L: Computer create couch (successful recognition)

R: [points to location to tell the system where the chair is to be created]

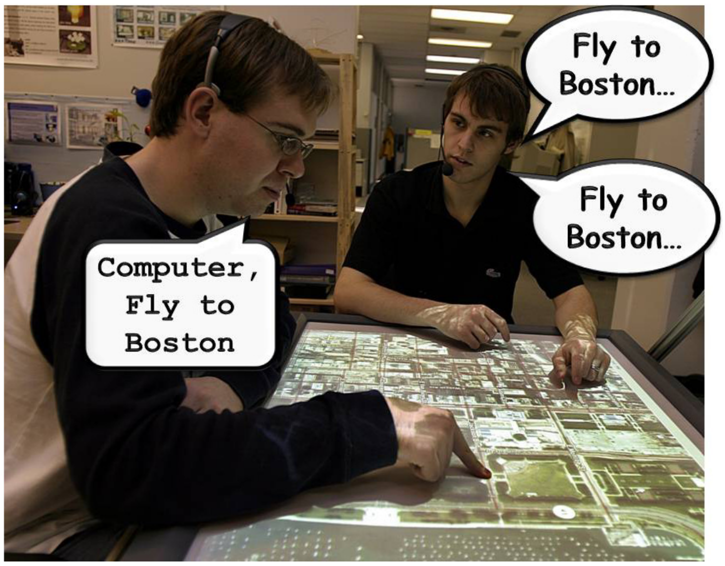


Figure 4.9 An illustration of cooperative error correction

Tightly Coupled, Intra-Modal. One person discusses or gestures over what should be done while the other person performs the command on the system. These interleaving acts were primarily used for two purposes. First, people used them to support coaching, validation and assistance. By suggesting what command should be performed next, participants are implicitly seeking validation of their suggestion from their partners. Second, this mode was used for cooperative error correction. In particular, when a person was having problems getting the system to recognize a particular speech or gesture command as valid input, the other person would often provide support by issuing the same command on their behalf. As seen in Figure 4.9, the right person might say "fly to Boston" twice not realizing that he was forgetting to say the "Computer" prefix. This could be corrected by their (left) partner saying "Computer, fly to Boston".

To digress momentarily, cooperative error correction within this mode is extremely important: it provides an additional level of robustness to multimodal systems. Previous empirical studies described how multimodal systems can add robustness; each modality provides a check for erroneous recognition [Oviatt, 1999]. For example, a "create stove" speech command would be ignored by GSI DEMO if no location-indicating gesture followed. Cross person error correction adds further robustness over this system correction. To illustrate, I recorded 84 speech recognition errors in the video transcriptions where the system failed to correctly recognize a speech command. Of these, partners stepped in ~1/3 of the time to correct another's error. Most participants would start by trying to reissue the command themselves. Two or more failed speech recognition attempts might be seen as an implicit request for assistance according to Clark's [1996] description of track two efforts to improve communication, and repair conversation.

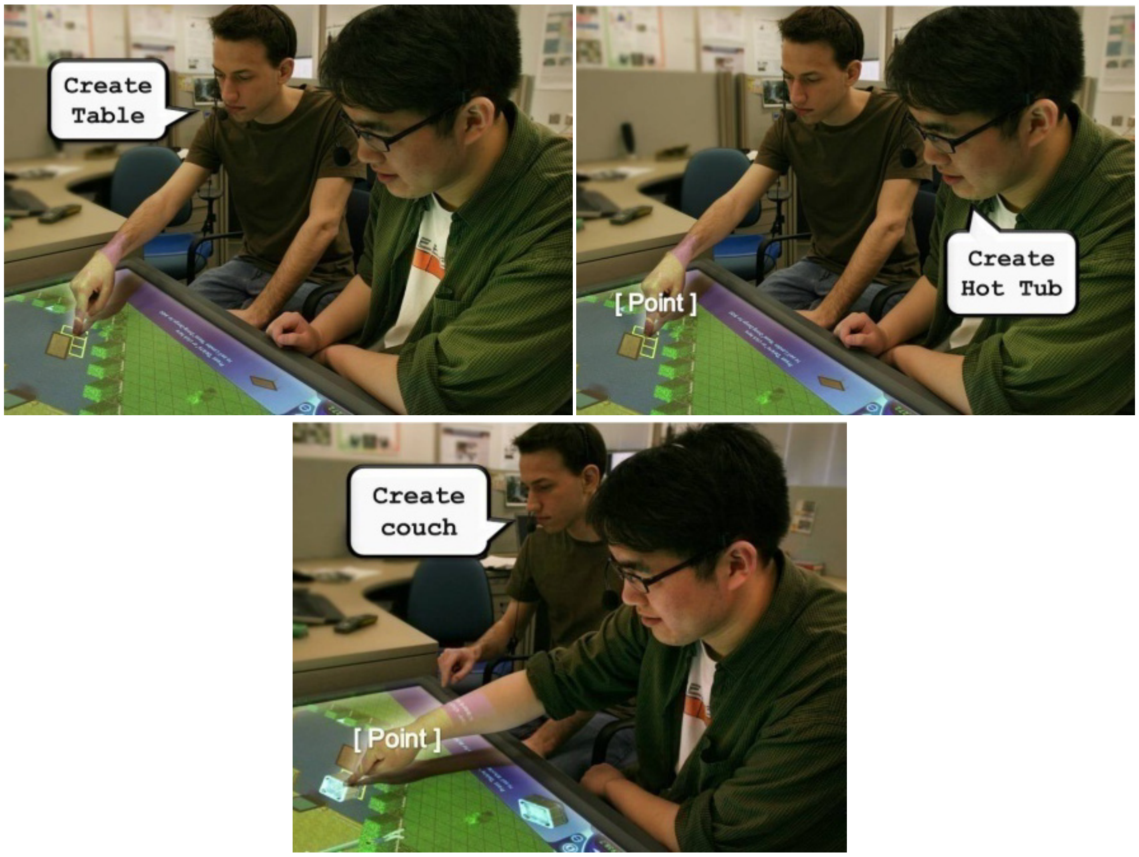


Figure 4.10 An inter-modal cross person multimodal action

Loosely Coupled, Inter-Modal. One person issues the next speech command while the other is finishing their gesture, i.e., they overlap command sequences, which the system then queues to the underlying single user application. This allowed pairs to efficiently issue overlapping multimodal commands without having to wait for the other person to finish their action. As illustrated in Figure 4.10, the left person could say the speech command "create table" (left) and while he was placing the table the right person could begin the next speech command "create hot tub" (middle). This process could be repeated while the right person placed the hot tub as illustrated in Figure 4.10 right. In practice, each participant peripherally monitored the workspace to find an appropriate place to insert their next command; they rarely overlapped commands in ways that resulted in system confusion.

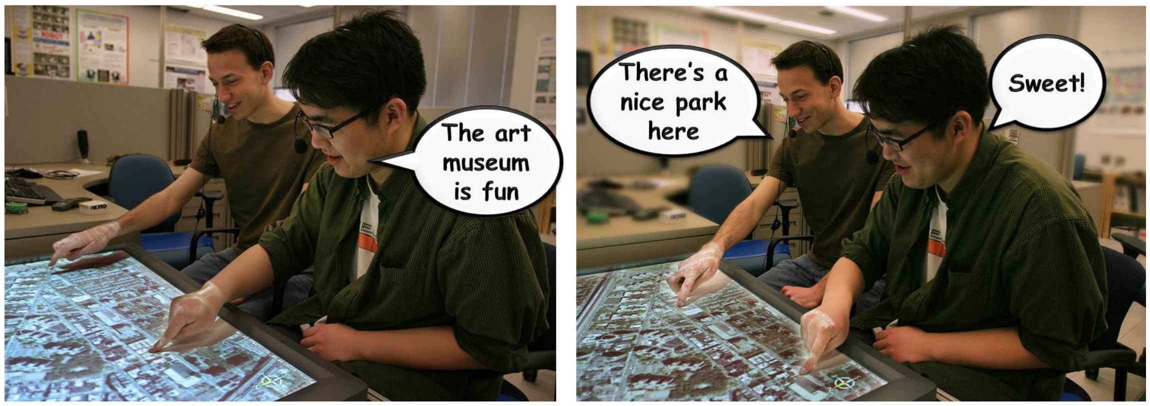


Figure 4.11 Interleaving Floor Control Actions

Loosely Coupled, Intra-Modal. One person issues a speech or gesture command within a conversation to assert informal floor control of not only the application, but of the conversational direction. For example in the travel planning task, people would often assert control of the map to signal that it was their turn to speak or to advance the discussion in a new direction. The other person would follow this lead. Figure 4.11 shows the right person panning the table towards an art museum (left), and then the left person interrupts this panning action by placing his finger on the table (right). By using a last gesture wins turn taking policy people can interrupt the actions of others to bring their attention to sights that might have otherwise been missed.

4.4 Conclusion

In this chapter I observed how pairs use collaboration-transparent speech and gesture wrappers over existing applications on a digital table. I saw that speech and gesture commands directed to the computer also served double duty as implicit communication to others (Figure 4.1). I saw that people's simultaneous interactions were not inhibited by the underlying single-user application. Similarly, I saw that people were able to compose sequential actions through interleaving acts: the graceful mixing of both participant's speech and gesture actions as commands were being constructed. All these are positive. They suggest that people can use multi-user multimodal tabletops - even when limited by single user application constraints – in much the same way as they work over visual work surfaces.

This chapter illustrates the behavioural benefits of using speech and gesture commands in a co-located setting, and validates much of the work presented in Chapter 3. The next chapter investigates the underlying architecture for enabling multi user speech and gesture interaction over a digital table.

Chapter 5. GSI DEMO: Multi User Gesture & Speech Interaction by Demonstration

***Objective Four**. I will develop a toolkit to enable rapid prototyping of multimodal co-located interactive systems.*

In previous chapters I implemented several speech and gesture wrappers over existing applications and saw that these multimodal commands served as both commands to the computer and as communication to collaborators. In this chapter, I detail the architecture of these systems by describing a tool that allows people to map speech and gesture commands on a digital table to keyboard and mouse actions understood by existing applications.[3]

In this chapter, I describe a multi user Gesture and Speech Interaction by Demonstration toolkit: GSI DEMO. First, GSI DEMO creates a run-time wrapper around existing single user applications: it accepts and translates speech and gestures from multiple people into a single stream of keyboard and mouse inputs recognized by the application. Second, it lets people use multimodal demonstration – instead of programming – to quickly map their own speech and gestures to keyboard/mouse inputs. For example, continuous gestures are trained by saying "Computer, when I do *[one finger gesture]*, you do [mouse drag]" (Figure 5.1). Similarly, discrete speech commands can be trained by saying "Computer, when I say layer bars, you do *[keyboard and mouse macro]*". The result is that people can rapidly adapt existing applications for use on a multimodal digital table. GSI DEMO also enables programmers to develop custom multimodal co-located applications.

[3] Portions of this Chapter are published as:

Tse, E., Greenberg, S. and Shen, C. (2006) GSI DEMO: Multi User Gesture / Speech Interaction over Digital Tables by Wrapping Single User Applications. Proc Eighth International Conference on Multimodal Interfaces (ICMI'06), (Nov 2-4, Banff, Canada), ACM Press, 76-83.

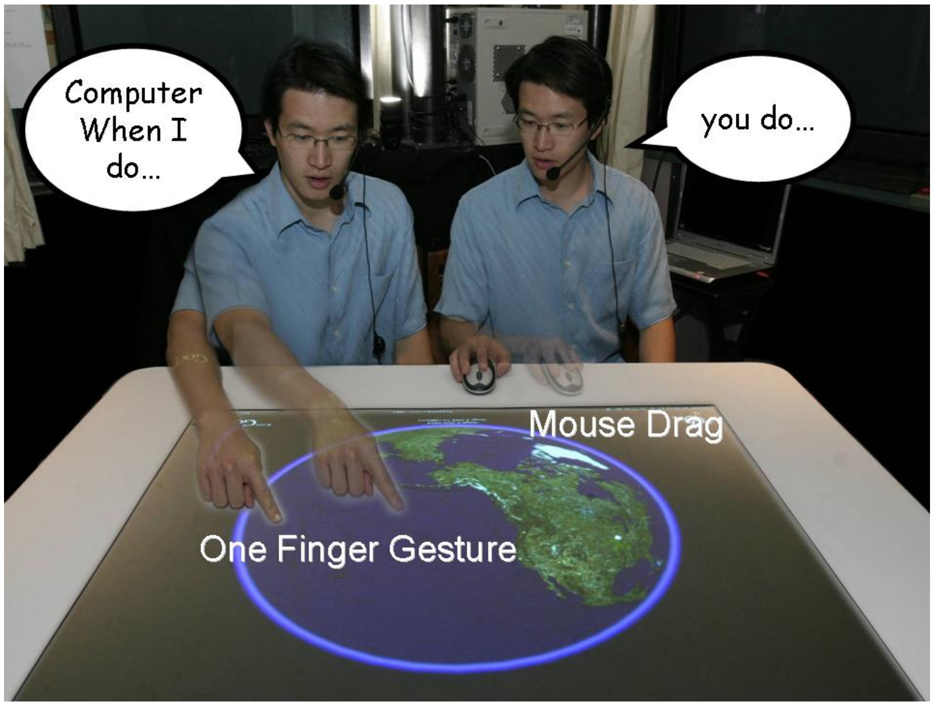


Figure 5.1. Using GSI DEMO

5.1 Introduction

A burgeoning research area in human computer interaction is digital table design, where natural table interaction is recreated and extended in ways that let multiple people work together fluidly over digital information. However, existing applications designed for a keyboard and a mouse interaction are limited when compared to the speech and gesture actions observed in real world face to face collaborations [Cohen *et al.*, 2002, McGee *et al.*, 2001]**.** A critical factor is that most digital systems are designed within single-user constraints. Only one person can easily see and interact at any given time. While another person can work with it through turn-taking, the system is blind to this fact. Even if a large high resolution display is available, one person's standard window/icon/mouse interaction – optimized for small screens and individual performance – becomes awkward and hard to see and comprehend by others involved in the collaboration [Gutwin *et al.*, 1998].

In this chapter I focus on the underlying architecture behind the systems described in my Thesis, which I call GSI DEMO - *Gesture and Speech Infrastructure created by Demonstration*. GSI DEMO offers:

1. Collaboration-transparent multimodal programming by demonstration;
2. A multi-user speech and gesture input wrapper around existing mouse/keyboard applications; and,
3. A software toolkit to support the rapid prototyping of collaboration-aware multi user multimodal groupware systems.

A key research contribution of GSI DEMO is to enable people to interact multimodally by demonstration – instead of programming – to quickly map their own speech and gestures to keyboard/mouse inputs (Figure 5.1). One trains the system by demonstrating the gestures and speech actions it should recognize, and then performing the appropriate keyboard and mouse events it should play back. For example, in Figure 5.1 we see a person train a single finger to left mouse mapping by demonstration. He begins by saying "when I do" dragging his finger across the table, and then saying "you do" and performing a drag action with a mouse and finally saying "okay". The result is that end users can quickly and easily transform single user commercial applications into a multimodal digital tabletop system.

This multimodal programming by demonstration relies on a run-time wrapper around existing single user applications. This wrapper accepts speech and gestures from multiple people working over the table, compares them to a list of allowable actions, accepts the ones that have a reasonable match, and then translates these into a single stream of keyboard and mouse inputs recognized by the application.

I begin with a high level overview of the GSI DEMO architecture. This is followed by a description of our gesture / speech recognizer and unifier. I then show how people map gesture and speech to application actions by demonstration, and how these are invoked after recognition. I then describe how a programmer can use GSI DEMO to develop collaboration-aware multi user multimodal applications. I close with a brief discussion of the strengths and limitations of this approach.

5.2 GSI DEMO Overview

As mentioned, GSI DEMO is designed to allow end users to rapidly create, and use their own multimodal gesture/speech input wrappers over existing single user applications (as shown in Chapter 3). However, it is not a generic gesture engine or speech recognizer. I stress that

GSI DEMO is not an advanced speech or gesture recognition system as I use off the shelf products and algorithms. The focus of this research is to leverage speech and gesture recognition for interaction by multiple people in a co-located environment. GSI DEMO will benefit from the more robust and reliable speech and gesture recognition systems of the future. Its main capabilities are summarized below.

Gesture recognition. GSI DEMO focuses on gestures that model the basic everyday acts seen in tabletop use, e.g., [Cohen *et al.*, 2002, McGee *et al.*, 2001, Tse *et al.*, 2006]. Examples include one-finger pointing for selection, dragging and panning, two-handed multiple object selection by surrounding an area with upright hands [Wu *et al.*, 2006], two finger stretching to zoom in and out of a region, fist stamping to create new objects, palm-down wiping to delete objects. These gestures have to be easily understood by others, as they also serve as communicative acts (as observed in Chapter 4). Consequently, GSI DEMO does not readily support complex and arbitrary gestures that act as abstract command surrogates, e.g., a D-shaped gesture over an object to indicate 'Delete'.

Speech recognition. GSI DEMO emphasizes the recognition of simple discrete speech utterances that match a fixed vocabulary of commands (e.g., 'fly to Boston', 'stop'…). Continuous speech is monitored for occurrence of these utterances.

Speech in tandem with gestures. GSI DEMO recognizes combinations of speech and gestures, where gestures can qualify a person's speech acts. For example, a speech command 'Create a tree' can be followed by a [point] gesture that indicates where that tree should be created.

Floor control. GSI DEMO also recognizes that near-simultaneous gestures and speech performed by multiple people may be combined, interleaved and / or blocked. It does this by supplying a set of appropriate floor control mechanisms that mediate how the system should interpret turn-taking.

Input surfaces. GSI DEMO is constructed in a way that handles multiple types of touch surfaces. Our current version handles the DiamondTouch Surface [Dietz and Leigh, 2001] and the DViT Smart Board [http://www.SmartTech.com]. Each device offers different input capabilities, thus allowing different (and not necessarily compatible) types of gestures. They also differ in how inputs from multiple people are disambiguated.

Multimodal training. People teach the system by demonstration, where they map their own speech and gestures to keyboard/mouse inputs recognized by the single user application.

As seen in Figure 5.2, GSI DEMO is roughly composed of three layers. The first layer contains all of the low level gesture and speech recognition systems. Each GSI Speech Client, one per person, recognizes speech commands spoken by each individual working over the table. The MERL DiamondTouch and the Smart DViT Gesture Engines recognize hand gestures done atop two different types of touch surfaces. The speech and gesture commands of multiple people are combined (with appropriate floor control mechanisms to mitigate problems arising from overlapping actions) in the GSI Gesture/Speech Unifier and converted into unique commands that will be used to activate the keyboard and mouse mappings. The Gesture and Speech Unifier is accessible to application programmers to assist the development of collaboration-aware multi user multimodal applications. The second layer defines the actual mapping of speech/gesture commands to actions understood by the single user application. Through the GSI Recorder, end users demonstrate multimodal speech and gestural acts, and then demonstrate the corresponding mouse and keyboard commands that should then be performed over that application. The Recorder generalizes these actions, and maps them together as a command. It then saves all that information to an application mapping file, where files can be created for different applications. For a given application, the GSI Player loads these mappings from the appropriate file. It then performs the corresponding keyboard and mouse commands when a speech or gesture command has been recognized by clients in Layer 1. Finally, the third layer is the single user commercial application that is oblivious to the recording and playback of multimodal speech and gesture actions of multiple people on the table top. From its perspective, it just receives mouse and keyboard input events as if they were entered by a single person.

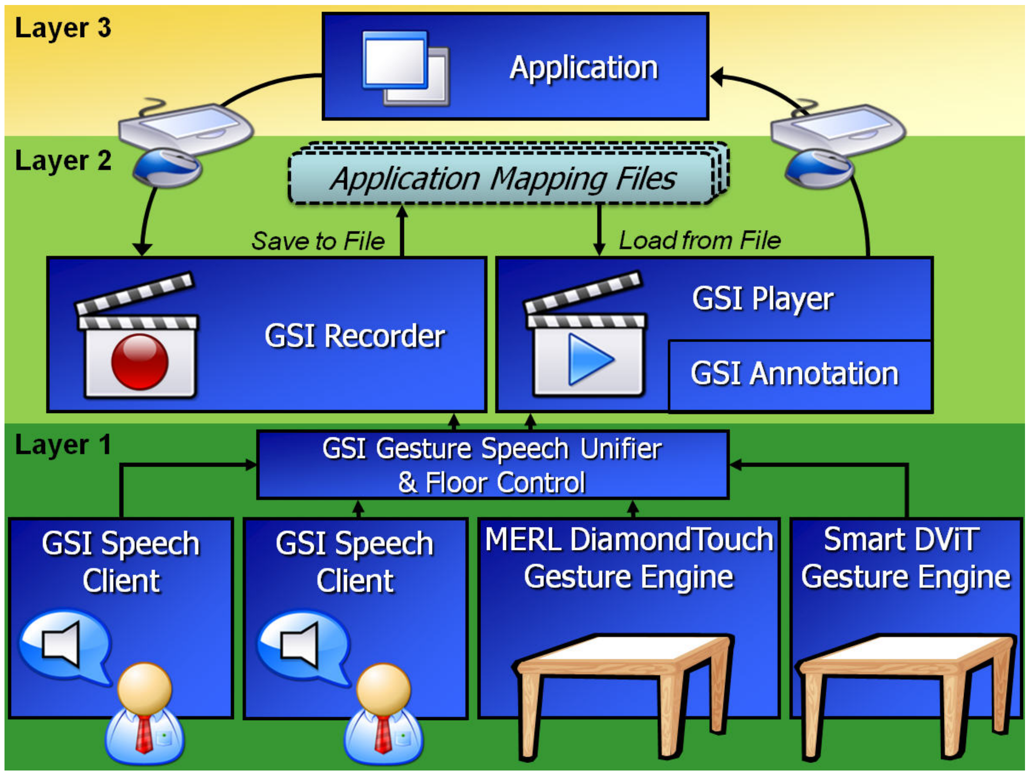


Figure 5.2. The GSI DEMO Framework

5.3 Speech / Gesture Engine

This section describes how the various components in Layer 1 work and interact. A description of end user programming can be found in Section 5.5.3.

5.3.1 GSI Speech Client

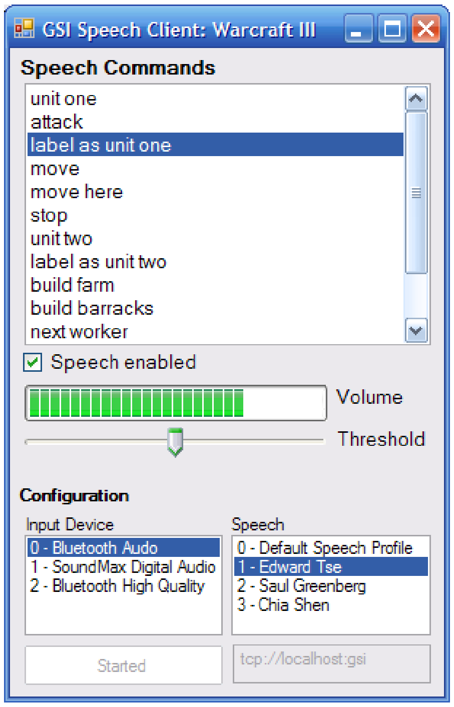
The GSI Speech client is responsible for speech recognition. As described below, it separates conversational speech from computer commands, disambiguates speech from multiple people, and has a few simple strategies for dealing with noise. Its output is a command, which is the best match of the spoken utterance against a list of possible computer commands.

Input hardware. Our speech recognition hardware uses a combination of high quality wired lavaliere microphones along with several wireless Bluetooth microphones. Microphones are noise-cancelling, so that only the speech of the person closest to the microphone is heard. Wireless microphones are definitely preferred, as this lets people walk around the table.

Speech recognition software. I use an existing off-the-shelf speech recognition system: the Microsoft Speech Application Programmers' Interface (Microsoft SAPI). However, SAPI and other similar off-the-shelf speech recognizers are designed to support one person using a computer in a relatively quiet environment. Co-located tabletop environments differ in 3 ways:

1. *The aural environment is noisy*. Multiple people may speak at the same time, and the constant actions of people around the table introduce additional sounds.
2. *Not all utterances are directed at the computer*. Most of the speech is actually normal conversation between participants.
3. *Multiple people may be trying to direct the computer*. We need to recognize and disambiguate speech from multiple people.

In light of these circumstances, I designed a specialized application around the Microsoft SAPI system called the GSI Speech Client, whose graphical interface is shown in Figure 5.3. Its job is to manage speech recognition gathered from multiple people working on the table display, where it delivers possible commands that these people are trying to invoke over the single user application. The GSI Speech Client (Figure 5.3) is typically shown on a secondary screen is not normally used by end users. It is usually used for debugging purposes as it reveals the speech recognition hypotheses of the system.



unit one
attack
label as unit one
move
move here
stop
unit two
label as unit two
build farm
build barracks
next worker

0 - Bluetooth Audo
1 - SoundMax Digital Audio
2 - Bluetooth High Quality

0 - Default Speech Profile
1 - Edward Tse
2 - Saul Greenberg
3 - Chia Shen

Figure 5.3. The GSI Speech Client GUI.

First, I simplify recognition issues in the presence of noise by giving each person a separate microphone headset; people select the microphone they are using from the Input Device list (Figure 5.3, bottom left). When a person speaks, the client matches that person's speech utterances against a dynamically reconfigurable list – a menu – of speech commands that should be understood (as described in the Application Mapping File, see GSI Recorder §5.4.1). This list is visible in Figure 5.3, top half. If the match is good, it offers that menu command as a working hypothesis of what the person has said. In contrast to free speech recognition, this simple matching of utterance to a list element is considerably more reliable in the presence of noise, and suits our purpose of matching speech to simple commands that can then be invoked.

Second, I disambiguate natural conversations from speech commands directed to the computer in three ways. Our first strategy uses microphones containing a switch; people turn it on when commanding the computer, and off when talking to each other. This is problematic because people forget, and because switching actions interfere with people's ability to gesture atop the table. Our second strategy is based on observations that people speak louder and clearer when issuing speech commands to a computer as compared to when they are speaking to each other across the table [Lunsford *et al.*, 2005]. I leverage this by providing a user-modifiable ***Minimum Volume Threshold*** (Figure 5.3 middle, current volume level is shown atop of the adjustable trackbar). Any talk below this threshold is considered interpersonal conversation, while talk above this threshold activates an attempt to recognize the current utterance as a command. Our third strategy is inspired from Star Trek, where actors direct commands to the computer by prefixing it with the key speech phrase 'Computer'. In GSI Speech, this phrase is configurable, although I too use the 'Computer' prefix. Thus 'Computer: Label as Unit One' is considered a speech command, while 'Label as Unit One' is conversation. Regardless of the strategy, the computer indicates recognition by highlighting the item on the list in Figure 5.3, and by playing a short sound; this audio feedback suffices to let people know that the computer has attempted to interpret their speech. In practice, the number of false speech command recognition errors is significantly reduced when a speech command prefix is used.

Third, I manage multiple people by supplying multiple independent speech recognizers. Most 'out of the box' speech recognizers operate as single-user systems, i.e., one cannot easily plug in multiple microphones and expect the speech recognizer to disambiguate between them. I have two strategies for managing this. One option is to use multiple computers, one per person, where each operates its own independent SAPI speech recognizer. The GSI Speech Client also runs on each computer, and collects all audio volume, hypothesis and speech recognition information. It posts this information to a distributed data structure seen by the main GSI DEMO client. The second option creates multiple speech recognizers on a single machine, where particular microphones are directed to particular speech recognizers. As before, information collected by each process is posted to a distributed data structure. The tradeoff between these options is CPU load (when multiple recognizers run on a single machine) and the requirement of multiple sound cards vs. the complexity and cost of having multiple computers. The GSI Speech Client is flexible enough so that one can mix and match, although I typically use separate computers to reduce CPU load. Regardless of the method chosen, the client lets a person select their custom speech profile previously created using Microsoft SAPI (Figure 5.3, bottom right).

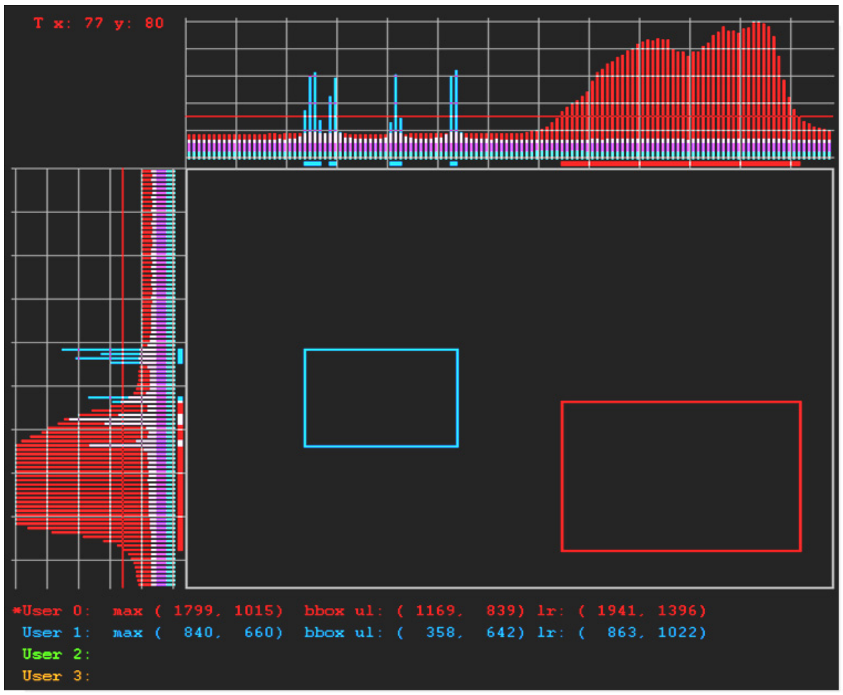
As mentioned, each speech recognition results in information posted to a distributed data structure. I use the GroupLab Collabrary to manage this sharing of information between processes and across multiple machines [Boyle, 2005]. The GroupLab Collabrary allows data to be posted to a hierarchical list of key/value pairs as illustrated below. For example, if field is filled then all connected machines receive a notification that a value in the list has been changed. Each client posts the following information for every utterance it recognizes:

```
/{ClientId}/Menu = "unit one, label as unit one, move here, ..."
/{ClientId}/SpeechEnabled = true
/{ClientId}/AudioLevel = 0 (min) to 100 (max)
/{ClientId}/Recognition = 1
```

The `ClientID` is a 32 bit integer dynamically defined at the beginning of program operation, which disambiguates who has spoken. Thus if there are three people using the table, there will be three different Client IDs. `Menu` is the complete command list; the system is restricted to matching an utterance against an element on this list. An important feature is that this command list can be dynamically changed at run time to load new command sets from the appropriate Application Mapping file (Figure 5.2). Thus once a Speech Client has been started, it does not need to be restarted to work with other applications and / or application modes. The `AudioLevel` gives a relative volume; decisions can then be made on whether to ignore speech uttered below a certain threshold. Finally, the client indicates what command it believes the person has said through the `Recognition` field; this is an index into the `Menu` command list. That is, the Recognition index of '1' in this example means that 'label as unit one' has been recognized. Finally, `SpeechEnabled` is true if that speech should be recognized. This could be toggled by an individual (as in Figure 5.3), or by a floor control policy (discussed later).

5.3.2 GSI Gesture Clients

GSI DEMO uses simple gesture recognition to interpret people's inputs on various touch surfaces. Currently, I use two different table input technologies: the MERL DiamondTouch and the DViT Smart Board. Because each input device has different characteristics that affect what can be recognized, I create specialized clients for each of them (Figure 5.2, bottom right). However, recognition is delivered to other components of GSI DEMO in a way that hides whether gestures are originating from a MERL DiamondTouch or a DViT Smart Board. Regardless of the technology used, recognition occurs by first training the system (see GSI Recorder §5.4.1), and then by matching the gesture against this generalized training set. For this later recognition step, I statistically analyze a gesture to produce a set of features, and then compare these features to those defining the different postures produced by GSI Recorder. If a good match is supplied, that posture is recognized and an event is raised. Details specific to each client are described below.



```
T x: 77 y: 80
*User 0:  max ( 1799, 1015)  bbox ul: ( 1169,  839) lr: ( 1941, 1396)
 User 1:  max (  840,  660)  bbox ul: (  358,  642) lr: (  863, 1022)
 User 2:
 User 3:
```

Figure 5.4. DiamondTouch Signal for Two People.

The MERL DiamondTouch Gesture Client. The MERL DiamondTouch, from Mitshubishi Electric Research Laboratories, affords multi-people, multipoint touches [Dietz *et al.*, 2001]. It uses an array of antennas embedded in a touch surface, each which transmits a unique signal. This technology provides an X and Y signal of each user's multi-point contact with the DiamondTouch surface. This produces a projection where the contact gradient provides useful recognition information. For example, Figure 5.4 graphically illustrates the signals generated by two people: the left bounding box and the top/side signals was generated by one person's 5-finger table touch, while the right was generated by another person's arm on the table. As each user generates a separate signal, the surface distinguishes between their simultaneous touches. For multiple touches by one person, there is ambiguity of which corners of a bounding box are actually selected. Currently, the DiamondTouch surface can handle up to four people simultaneously at 30 frames per second. Computer displays are projected atop this surface, and touch positions are calibrated to the computer display's coordinate system.

Again, I stress that my work is not about advancing the state of the art of gesture recognition, rather I use existing algorithms. In this case, I use a technique similar to [Wu *et al.*, 2006], I use this signal information to extract features, and to match these to different whole hand postures for any given instant in time, (e.g., a finger, the side of a hand, a fist). I then detect gestures: these are posture movements over time. Example gestures are two fingers spreading apart, or one hand moving left.

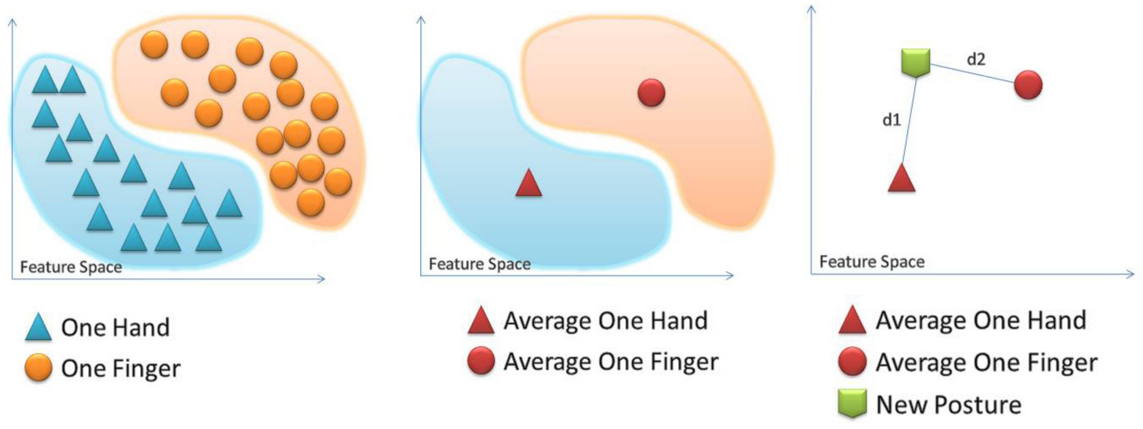


Figure 5.5 An illustration of the Univariate Gaussian Clustering algorithm

To recognize different gestures from this signal information I use a Univariate Gaussian Clustering algorithm [Duda *et al.*, 2000]. I statistically analyze the tabletop input to generate a list of 25 unique features (e.g., bounding box size, average signal value, total signal values above a particular threshold, etc). Different values of these features characterize a variety of different possible hand postures. When a hand posture is trained (see GSI Recorder §5.4.1) a list of values in feature space are recorded for each posture (Figure 5.5, left), the average value of each feature along with a numerical variation estimate is computed for each posture (Figure 5.5, middle) and saved to a file. During gesture recognition, the distance from the current posture to all other posture averages is calculated and the closest matching posture is determined (Figure 5.5, right). If it is within the numerical variation limits, a gesture recognition event is generated. If it closely matches two postures or is below a 'confidence' threshold, an unknown gesture event is raised. For ease of use a default gesture set is included that recognizes: one finger, two fingers, five fingers, one hand, one hand side (a chop), one fist and one arm. In practice, this clustering technique has proved to be quite effective across different hand sizes and levels of skin conductivity.

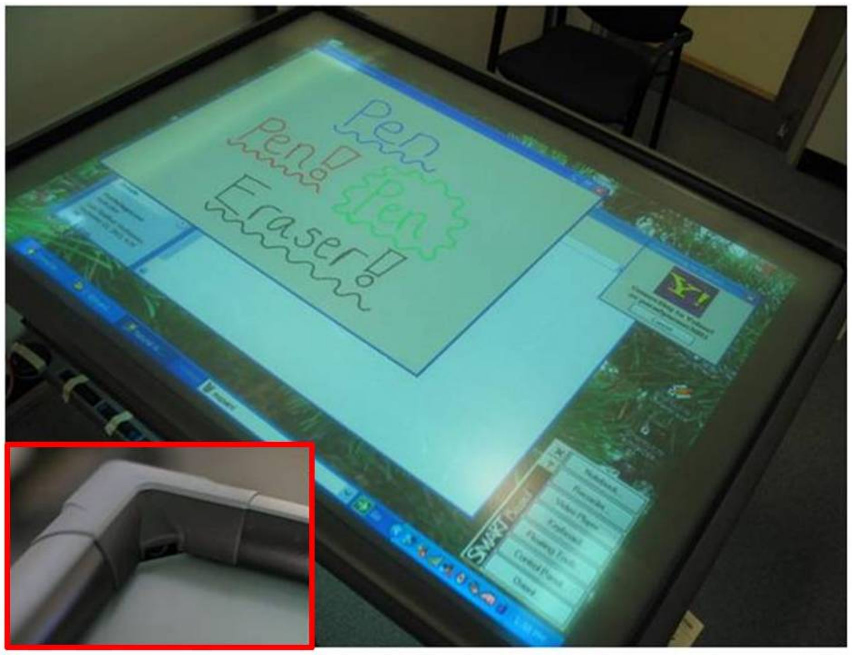


Figure 5.6. The DViT board: Four Infrared Cameras are located on each corner of the display. On each side of the display is an array of infrared LEDs

Smart Board DViT Gesture Client. The DViT Smart Board, from Smart Technologies Inc., uses four infrared cameras placed in each corner of the display (as illustrated in Figure 5.6, bottom left). This technology is capable of detecting up to two points of contact along with their respective point sizes. Infrared cameras produce a binary signal (on/off) as opposed to the signal gradient provided by the MERL DiamondTouch. Thus it is somewhat more difficult to recognize particular hand postures (e.g., hand *vs.* chop *vs.* fist). Currently, the technology does not distinguish between different users.

In practice, the point size returned by the DViT can be used to determine the difference between a pen tip, a finger, a hand and a whole arm. However, similarly sized hand postures (e.g., one hand, one fist, one chop) currently cannot reliably be distinguished on the DViT. As well, it cannot identify which person is doing the touch. For now, our DViT Gesture Client recognizes a subset of the DiamondTouch Gesture Engine Events, and all gestures appear as if they are originating from a single person.

5.3.3 GSI Gesture/Speech Unifier

The next step is to integrate speech and gestural acts. This is done through the GSI Gesture Speech Unifier (Figure 5.2, Layer 1).

All speech and gesture recognition events are sent to a speech and gesture unifier that is composed of two parts: a single user speech and gesture unifier and a multi-user floor control unifier.

Single user speech and gesture unifier. In our system, speech and gestures can be independent of one another, e.g., when each directly invokes an action. Examples are 'Fly to Boston' in Google Earth, which directs the system to navigate and zoom into a particular city, or a wiping gesture that immediately erases marks underneath the palm. Speech and gesture can also interact as a multimodal command. For example, a person may say 'Create a table' in The Sims, and then touch the spot where the table should be created. In this later case, the gesture/speech unifier must create and deliver a well-formed command to the single-user application from the combined gesture and speech elements.

Our unifier uses Johnston *et al.*'s unification based multimodal integration [1997]. If the speech component of a multimodal command is recognized, it must be matched to an appropriate gesture within a certain time threshold, or else the command is treated as a false recognition. In this way it is possible for multimodal speech and gesture to provide more reliable input than speech or gesture alone [Oviatt, 1999]. The multimodal unification time threshold can be dynamically adjusted by the end user to suit their multimodal preferences. In practice, a threshold of a second or two suffices.

Floor control. Contention results when multiple people interact simultaneously over current single user commercial applications, where each supplies conflicting commands and/or command fragments. For example, if two people try to move an application window simultaneously the window will continuously jump from the hand position of the first user to the hand position of the second user. To mitigate these situations, the Gesture/Speech unifier contains various multi-user floor control policies to mediate turn-taking and how people's actions are interleaved with one another. As shown in Figure 5.8 (bottom), a person can select the particular multi user floor control policy using a graphical interface.

The output of each single user speech and gesture unifier is delivered to the current multi-user floor control policy module. This module collects and filters the output based on a particular set of rules. The simplest policy is *free for all,* where the system tries to interpret *all* speech and gestures into well-formed commands. Contention avoidance is left up to the group's social practice. The advantage is that people can interleave their actions, while the disadvantage is that accidental overlap introduces confusion and/or undesired application responses. Other policies balance contention avoidance with social practice. In the case of competing gestures, the *last gesture and last speech wins* policy only listens to the person who had the most recent gesture down event, ignoring the overlapping speech and gesture movements of other users. Our *micro turn-taking* floor control policy adds to this by letting one's gesture acts complete the multimodal speech command of another person. For example, one person can say 'Create a tree', and others can indicate where it should be created. This allows the group to distribute the decision making process to all seated around the table. Other policies (e.g., first gesture or first speech wins) enforce stricter turn-taking, where others are blocked until the current person has completed a well-formed gesture/speech command.

The organization of the GSI Speech/Gesture Unifier makes it easy to create new floor control policies. Programmers of such policies only deal with high level speech and gesture recognition events rather than low level speech and gesture APIs. They do not need to know if events are generated by the DiamondTouch, the DViT, or even other technologies that could be added in the future.

5.4 Speech/Gesture by Demonstration

For GSI DEMO to work with single user applications, it needs to somehow map speech and gestures into actions understood by that application. That is, it needs to know:

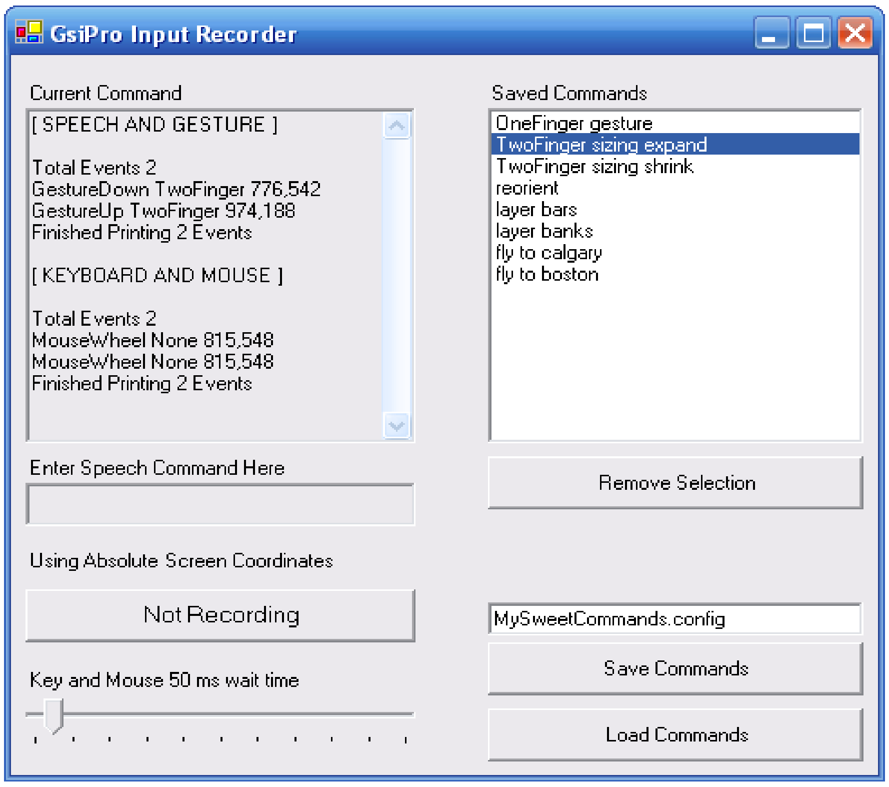
1. *The command set.* What commands and arguments should be used to interact with the single user application;
2. *The speech and gesture commands.* What gestures and speech correspond to those commands and their arguments;

3. *The keyboard and mouse acts.* How to invoke application actions when a command is invoked (e.g., mouse down/move/up, menu or palette selections, keyboard entry).

While this information could be hard-wired into the application, GSI DEMO lets all this to be configured on the fly by demonstration (Figure 5.2, layer 2). The idea is that people use the GSI Recorder module to perform a speech / gestural action, and then follow this with the corresponding action on the single user application. The system generalizes the sequence, and packages this up as a macro. When a speech / gesture action is recognized, that macro is invoked by the GSI Playback module. While simple, the result is that end-users can quickly repurpose single user applications for multi-user, multimodal digital tabletops. The sections below describe how this is achieved.

5.4.1 GSI Recorder

The GSI Recorder, shown in Figure 5.7, lets end-users train the system on how to map speech, postures and gestures to actions understood by the single user application. However, before the Recorder is started, it must have two things. First, speech recognition needs a speech profile customized for each person; as mentioned I use the Microsoft SAPI system to generate these individual profiles. Second, a set of possible postures is needed, as people's gestures will be matched to this set. To train a posture, a person places a posture (e.g., a fist) on the surface, and then moves it around to various positions and angles over the surface for a short time. Instances of this posture are generated 30 times a second. As mentioned previously, statistics are performed over the raw signal or point information defining each instance to convert them into a number of features (e.g., the average signal in the bounding box surrounding the posture). I previously described how features across these instances are then generalized using a Univariate Gaussian Clustering algorithm to produce a description of the posture, saved to a file, and later used by the Recorder.



```
[ SPEECH AND GESTURE ]

Total Events 2
GestureDown TwoFinger 776,542
GestureUp TwoFinger 974,188
Finished Printing 2 Events

[ KEYBOARD AND MOUSE ]

Total Events 2
MouseWheel None 815,548
MouseWheel None 815,548
Finished Printing 2 Events
```

```
OneFinger gesture
TwoFinger sizing expand
TwoFinger sizing shrink
reorient
layer bars
layer banks
fly to calgary
fly to boston
```

Using Absolute Screen Coordinates

Key and Mouse 50 ms wait time

MySweetCommands.config

Figure 5.7. The GSI Recorder GUI

Training. Three different types of speech/gesture to keyboard/mouse mappings are supported by the Recorder: *continuous gesture commands* (e.g., a one hand pan is remapped onto a mouse drag), *discrete speech commands* (e.g., "fly to Boston" is remapped onto a palette selection and text entry sequence) and *multimodal speech and gesture commands* (e.g., "move here *[pointing gesture]*" is remapped onto a command sequence and mouse click).

Continuous gesture commands are trained by saying "Computer, when I do [gesture] you do [mouse sequence] okay". To break this down, GSI Recorder will attempt to recognize a hand gesture after the person says "Computer, when I do". If it does, it provides visual feedback by printing out the recognized gesture in the current command textbox (Figure 5.7, top left) and auditory feedback in the form of a light chirp. The person continues by saying "you do" and then performs the keyboard/mouse sequence over the single user commercial application. Because the application is live, the person receives instantaneous feedback about the status of the mouse sequence; abstracted actions are also displayed graphically (Figure 5.7, top left). The person says 'okay' to end the keyboard and mouse sequence. After the gesture is recorded, people can instantly test what was learnt by performing the appropriate gesture on the tabletop. If it is wrong, they delete it (the 'Remove Selection' button) and try again.

Discrete speech commands are trained similarly. The person says "Computer, when I say <*speech command*> you do *[keyboard and mouse sequence]* okay". Speech commands are typed in rather than spoken; later, the speech recognizer will parse spoken speech – regardless of who said it – to see if it matches the textual string. This is important, as it means that one person can train the system and others can use it. As before, discrete commands can be played back by saying the recently recorded speech command.

Multimodal speech and gesture mappings are created by saying "Computer, when I say <*speech*> *[gesture]* you do *[keyboard and mouse sequence]* okay". Gesture can also precede speech if desired. Again, speech commands can be quickly played back by performing the appropriate sequence of events.

Each successful mapping is recorded to a list box (Figure 5.7, right) that stores all successful commands. Clicking on a command reveals the recorded gesture and its corresponding keyboard and mouse sequence (Figure 5.7, left). The 'Save Commands' button stores the entire command set and its mapping between speech/gesture and keyboard/mouse actions to an Application Mapping File (Figure 5.2). The GSI Player will later load and use this file.

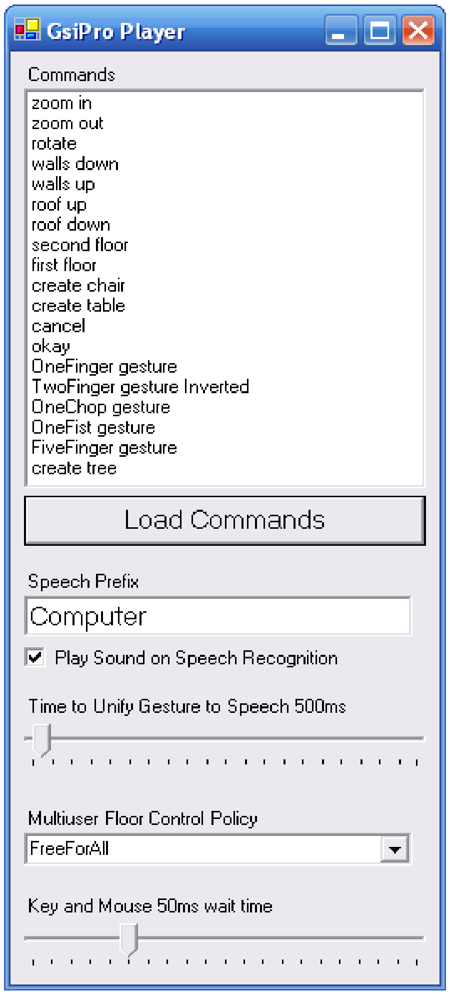
Generalization. Under the covers, GSI Recorder uses the Microsoft Windows Hook API to listen to the keyboard and mouse events of all applications while the end user interacts with the commercial application. However, it would be inappropriate to just record and playback this literal stream. Some generalization is needed.

The main generalization is how recognized gestures are translated into mouse events. For each gesture, the Recorder records its 'Gesture Down' and 'Gesture Up' event (along with its coordinates, posture, bounding box, and so on). It does *not* record all intervening 'Gesture Move' data. During later replay, it matches gestures to mouse coordinates by first taking the center of the posture as the mouse down, then the intermediate gesture points as the mouse move, and then the final gesture up point as the mouse up. Of course, this generalization means that movement gestures (e.g., a one finger circle) cannot be recognized.

I also found that there are times when it is inappropriate for the direction of a continuous gesture command to directly map onto the click coordinates of the corresponding mouse command. When the middle mouse button in some applications is held down to scroll, moving the mouse down (closer to the user) advances the document away from the user. In a gestural interface, the document would appear to be moving in the opposite direction than expected. GSI Recorder recognizes these cases. If a person specifies a gesture in one direction, but then moves the mouse in the opposite direction, it will invert the input appropriately.

Another generalization is that the literal timing information in keyboard and mouse moves is discarded. During replay, this has the advantage that long sequences of keyboard/mice actions are replayed quickly. As well, if sequences raise and then hide menus, tool palettes or dialog boxes, these pop-up screen elements may appear as a brief flash, or not at all. This gives the end users the illusion that the executed keyboard and mouse sequence is invoked as a single command. For example, the "layer bars" command in Google Earth is really a complicated sequence of opening the layer menu, toggling the bars option and closing the layer menu. This command appears as a brief 0.1 second flicker, with the end result of bars being layered on the digital map. However, I also recognize that some applications do need delays between events (e.g., time for a menu to fade away); in these cases, people add in explicit wait times using the "wait" speech command and they can use the 'Key and Mouse wait time' trackbar (Figure 5.8, bottom) to vary wait times.

Finally, another generalization considers the screen coordinates of mouse actions. Mouse events are normally received in absolute screen coordinates. If the application window is moved, or if interaction on a popup window is required, or if the screen resolution changes (thus shifting how some windows are arranged on the screen), the mapping will not work. One partial solution lets the user specify that actions are relative to a particular window by saying: "Computer, relative to window *[select window]*". GSI Recorder then uses the GroupLab.WidgetTap toolkit [Greenberg *et al.*, 2002] to find the string uniquely identifying that window. When the command is later executed, GSI Player will then search for the appropriate window and then make all coordinates relative to its top left corner.



Commands
zoom in
zoom out
rotate
walls down
walls up
roof up
roof down
second floor
first floor
create chair
create table
cancel
okay
OneFinger gesture
TwoFinger gesture Inverted
OneChop gesture
OneFist gesture
FiveFinger gesture
create tree

Speech Prefix: Computer

[✓] Play Sound on Speech Recognition

Time to Unify Gesture to Speech 500ms

Multiuser Floor Control Policy: FreeForAll

Key and Mouse 50ms wait time

Figure 5.8. The GSI Player GUI.

5.4.2 GSI Player

GSI Player loads the set of speech/gesture to keyboard/mouse mappings from the Application Mapping file (Figure 5.2, top), which is displayed in the Player's interface (Figure 5.8, top). Different files can be loaded for different applications, or even for different modes within an application. For example, I generated different mapping configurations for two variations of Google Earth, for The Sims, and for Warcraft III. When a file is loaded, the appropriate speech command set is sent to each GSI Speech Client, and the GSI Gesture/Speech Unifier is set to monitor appropriate speech and gesture commands as specified in the file. To the end user it appears as if the system now seamlessly accepts speech and gesture multimodal commands from that command set.

GSI Player lets people set a variety of other options (lower half of Figure 5.8):

- what floor control policy should be used (via the 'Multiuser Floor Control Policy' dropdown list);
- maximum time that must pass between speech and gestural acts if they are to be unified;
- whether a speech command prefix is to be used to enable utterance recognition, and if so what that should be (done by altering the text in the 'Speech Prefix' textbox
- toggle auditory feedback that happens when a command is recognized ('Play Sound on Speech Recognition' box).

5.4.3 GSI Annotation

GSI DEMO also automatically supports multi-user annotation over the single user application through its 'Scratch Pad' speech command. Using the GroupLab.Collabrary [Boyle *et al.*, 2005], the GSI Annotation captures an image of the primary screen, and then overlays this image on top of the commercial application (using a top level window). To the end user, it appears as if the application has 'frozen'. Multiple users can then simultaneously annotate with different colors per user using a single finger. Any posture larger than a single finger is treated as an erasing gesture, where erasure restores the underlying background image. Saying 'Scratch Pad' a second time returns to the original single user application. The collaborators see the marks disappear, and the application comes alive again.

5.5 Using GSI DEMO

This section provides step by step instructions of how an end user would use GSI DEMO. These examples illustrate how the GSI DEMO components work together for actual use. We begin with some tasks that are normally done before using GSI DEMO, then I describe how to map Mozilla Firefox for use on a multimodal tabletop. Finally, I describe how an end programmer would create their own collaboration-aware groupware applications using GSI DEMO.

5.5.1 Getting Started

Before using GSI DEMO it is helpful to first set up the GSI Speech Clients and to train any new hand postures using the MERL DiamondTouch Gesture Engine. Under normal circumstances the default set of hand postures is sufficient for many applications and will not need to be trained. Also, the GSI Speech Client and Shared Dictionary instances only need to be started once when the computer is booted (these instructions can be automated to run on computer start up). The instructions are included below:

Setting up the GSI Speech Clients

1. Train a Speech profile using the Microsoft Speech Control Panel for each person using the system. For multiple computer speech recognition, the Microsoft Speech Profile manager can be used to transfer speech profiles across computers.
2. Using the GroupLab Collabrary, create a Shared Dictionary instance called gsidemo on the machine that is connected to the tabletop. This is where all speech commands will be updated.
3. Now open the GSI Speech Client (Figure 5.3). A speech client can be opened on separate computers by adjusting the appropriate IP address before starting the speech recognizer. GSI Speech client can save the IP address and speech profile settings for

easy opening in the future. When opened a number of default speech commands appear in the list box, they can be spoken to test the speech recognizer accuracy.

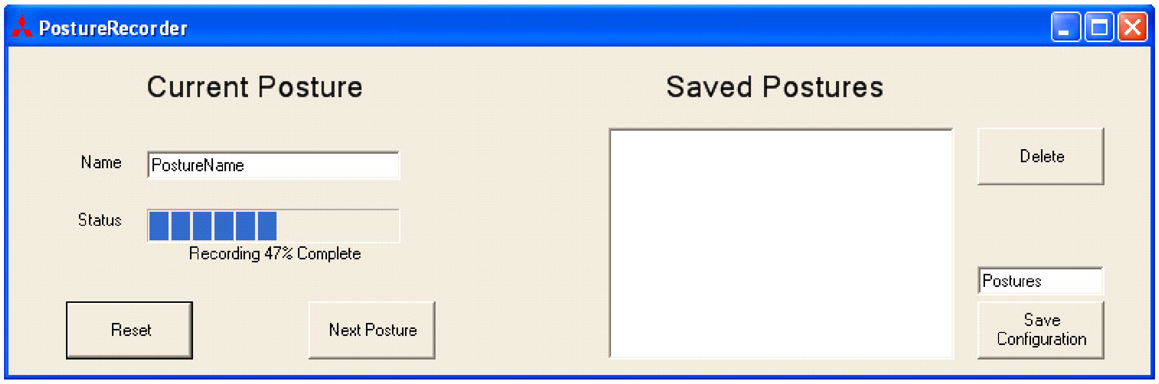


Figure 5.9 The MERL DiamondTouch Posture Recorder GUI

Training new hand postures using the MERL Gesture Engine

This step is not normally required as people can use pre packaged gestures, however, for completeness we show how people can train new hand postures.

1. Open the MERL DiamondTouch Posture Recorder (show in Figure 5.9)

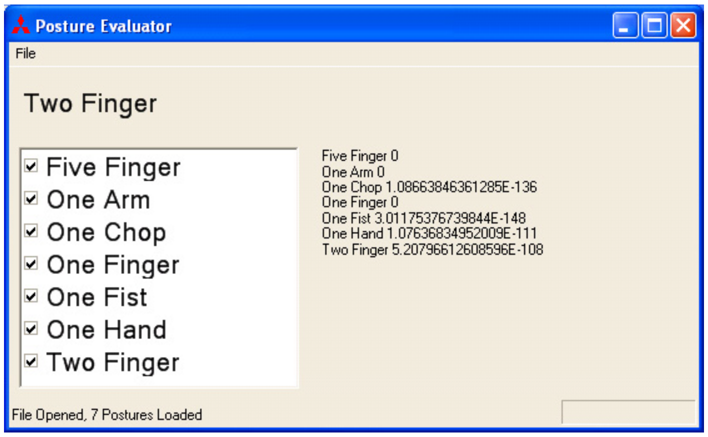
2. Place a new hand posture on the DiamondTouch table until the status bar reaches 100%. This provides a sufficient amount of feature samples to provide a good average and variation for each posture. Typically at least 1800 samples are captured over one minute for each hand posture.

3. When the posture has finished recording type the name of the posture (Figure 5.9, left) and press the Next Posture button. The current posture will be saved to the list box (Figure 5.9, right). Repeat steps 2 and 3 until all hand postures have been recorded.

4. When all hand postures have been recorded type a posture configuration name into the save configuration text box (Figure 5.9, right). If you wish to make this set the default configuration specify "default" as the configuration file name.

5. You can examine the effectiveness of the system to recognize different hand postures using the MERL DiamondTouch Posture Evaluator (Figure 5.10). When a configuration file is loaded the possible postures it can recognize are shown in the list box (Figure 5.10, left). By using different hand postures on the DiamondTouch table one can see if the recognized posture (the large text label in Figure 5.10, top) matches

what was done on the table. If the recognized posture is not consistent for a particular hand configuration, this usually indicates that the two hand postures are too similar. Hand postures recognition can be enabled and disabled by clicking on the check box to the left of each loaded posture. Any corrections to a hand posture profile must be done using the MERL DiamondTouch Posture Recorder.



Two Finger

- ✓ Five Finger
- ✓ One Arm
- ✓ One Chop
- ✓ One Finger
- ✓ One Fist
- ✓ One Hand
- ✓ Two Finger

```
Five Finger 0
One Arm 0
One Chop 1.08663846361285E-136
One Finger 0
One Fist 3.01175376739844E-148
One Hand 1.07636834952009E-111
Two Finger 5.20796612608596E-108
```

File Opened, 7 Postures Loaded

Figure 5.10 The MERL DiamondTouch Posture Evaluator GUI

5.5.2 Mapping a Web Browser

In this example I show how an end user could map an off the shelf Web Browser (Mozilla Firefox 1.5) using the programming by demonstration features of GSI DEMO. I illustrate how to map continuous gesture commands, discrete speech commands, inverted gestures, and bimanual selections.

1. Make sure that all speech profiles and appropriate hand postures have been trained as described in the previous Section (§5.5.1).

2. Open Mozilla Firefox and the GSI Recorder (Figure 5.7).

3. **Continuous Gesture**. We will start by mapping a fist to the middle mouse button. In Mozilla Firefox, the middle mouse button pans a web document. Say "Computer when I do" and drag a fist along the application, then say "you do" and middle click along the same area. Finally, say "okay" this should create our first mapping. You should see a new saved command called "One Fist gesture" in the command listbox (Figure 5.7, right). This causes the web document to move in the opposite direction of

the gesture. For example, a fist moved towards a user will move the document further away from the user.

4. **Inverted Gesture**. In this step, we will map an inverted gesture of five fingers to a middle mouse button to avoid the inverted document scrolling mapped in Step 3. Start by saying "Computer when I do" and then drag a five finger posture from the middle of the screen to the bottom. Then say "you do" and then drag your mouse from the middle of the screen to the top and say "okay". You should see a new saved command called "Five Finger gesture inverted" in the recorded commands list box (Figure 5.7, right). Inverted gestures are detected by looking at the position of the up and down events for both the gesture and mouse commands. In general, GSI DEMO ignores all gesture dynamics and only uses information found in the up and down events.

5. **Bimanual Selection Gesture**. Now we are going to map a bimanual selection to a click drag event. Say "Computer when I do" and then place one hand side on the table, wait a second and place the second hand side on the table and then remove both hands simultaneously. Now say "you do" and click and drag over the exact area that was selected between your hands. A new saved command called One Chop Bimanual Selection should be added to the recorded commands list box (Figure 5.7, right).

6. **Discrete Speech Command**. We will map the three basic navigational commands (home, back, forward) to speech actions. Say "Computer when I say" and type "home" then say "you do" and then use the keyboard shortcut alt-home. Repeat this procedure for "back" (alt-left arrow), and "forward" (alt-right arrow).

7. Now we've finished our mapping for FireFox, click on the Textbox above Save Commands and type in a file name FireFox then press the save command (Figure 5.7, bottom right).

5.5.3 Creating a collaboration-aware multi user multimodal application

This example application shows how a programmer might create their own collaboration-aware multi user multimodal application using a high level programming language (Microsoft C#). We will create a full screen colour mixing application (Figure 5.11) that will allow multiple people to draw on the tabletop white also being able to use speech commands ("red", "green", "blue", "yellow") to change the colour of their drawings. This example shows how to use unified speech and gesture input (as shown in Row 1 of Figure 5.2). It describes how an end programmer might use speech and gesture input in their own drawing application.

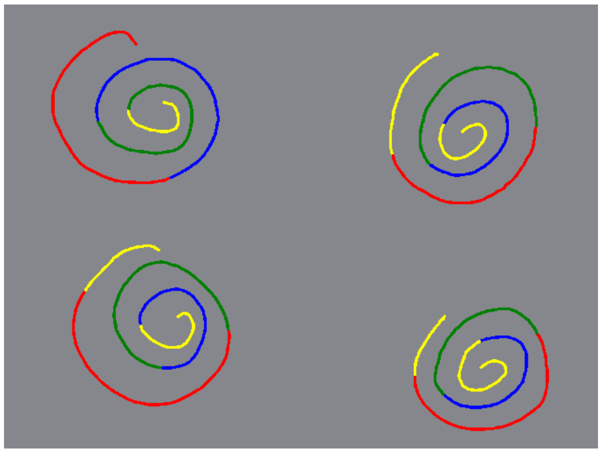
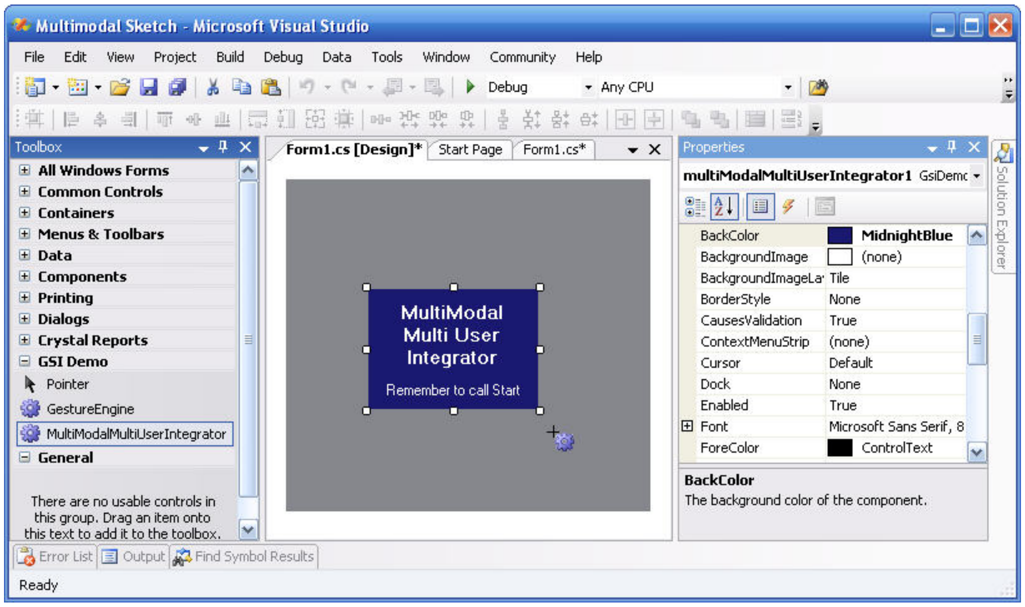


Figure 5.11. The Multi User Multimodal Colour Mixer Application

1. Make sure that all speech profiles and appropriate hand postures have been trained as described in the previous Getting Started Section (§5.5.1).

2. Start Microsoft Visual Studio 2005 and create a new C# project called "Multimodal Sketch".


Multimodal Sketch - Microsoft Visual Studio
File Edit View Project Build Debug Data Tools Window Community Help
Debug
Any CPU
Toolbox
All Windows Forms
Common Controls
Containers
Menus & Toolbars
Data
Components
Printing
Dialogs
Crystal Reports
GSI Demo
Pointer
GestureEngine
MultiModalMultiUserIntegrator
General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
Form1.cs [Design]*
Start Page
Form1.cs*
MultiModal Multi User Integrator
Remember to call Start
Properties
multiModalMultiUserIntegrator1 GsiDemo
Solution Explorer
Error List
Output
Find Symbol Results
Ready

BackColor	MidnightBlue
BackgroundImage	(none)
BackgroundImageLa	Tile
BorderStyle	None
CausesValidation	True
ContextMenuStrip	(none)
Cursor	Default
Dock	None
Enabled	True
Font	Microsoft Sans Serif, 8
ForeColor	ControlText

BackColor
The background color of the component.

Figure 5.12 Creating a Multimodal Sketching Application using Microsoft Visual Studio

3. You should see a Multi User Multimodal Integrator in the Visual Studio Toolbox. Click on that component and drag an instance onto the form. Your screen should look like Figure 5.12.

4. Now click on the event tab of the properties window (the lightning bolt) and double click on the PostureDown event handler. This will automatically generate a Posture Down event stub in the code window. Repeat step 4 for the GestureMove, GestureUp and SpeechRecognized events.

5. We are now ready to start coding. Switch to the code tab (called Form1.cs). And add the following initialization code:

```
//Included Libraries
using AxDIAMONDTOUCHLib;
using GestureEngine;
using GsiDemo;

//Variables used in Colour Mixer Application
private Color[] OriginalColors = {Red, Green, Blue, Yellow };
private Color[] UserColors = { Red, Green, Blue, Yellow };
private Point[] LastPoint;
private int TotalUsers;

public Form1() { //GSI Demo Initialization
     InitializeComponent();
     gsiDemoIntegrator.Start();
    gsiDemoIntegrator.Started += new EventHandler(GsiDemoStarted);
}
```

6. After the integrator has started we need to initialize the LastPoint variable and set the speech commands.

```
private void GsiDemoStarted(object sender, EventArgs e)
{
     TotalUsers = gsiDemoIntegrator.GetTotalGestureClients();
     LastPoint = new Point[TotalUsers];
     for (int i = 0; i < LastPoint.Length; ++i)
     {
          LastPoint[i] = new Point(-1, -1);
     }
     gsiDemoIntegrator.SetSpeechCommands("red pen,green pen,
                         blue pen,yellow pen,clear screen,");
}
```

7. Next, we will add some speech event handling code. Go to the SpeechRecognition event handler, the following code plays a sound cue after each recognized speech event and sets the appropriate colour, or clears the screen.

```
private void SpeechRecognition(int UserId, int SpeechIndex)
{
   //play an audio cue
   SoundPlayer sp = new SoundPlayer("audiocue.wav");
   sp.Play();

   if (SpeechIndex < 4)
       UserColors[UserId]= OriginalColors[SpeechIndex];
   else //clear the screen
       this.CreateGraphics().Clear(this.BackColor);
}
```

8. Now, we are ready to implement some basic drawing code, we will start with the Posture Down and Gesture Up events. We use Posture Down instead of Gesture Down because Posture Down is activated the moment that a finger touches the table while Gesture Down is only activated once a gesture has been successfully recognized over several input frames. When someone touches on the table we will set the Last Point to a new point and when it is released we will reset that variable to its default value.

```
private void PostureDown(int PostureDetected, TouchEvent e)
{
    LastPoint[e.ID] = new Point(e.x, e.y);
}

private void GestureUp(GestureEventArgs gesture, TouchEvent e)
{
   LastPoint[e.ID] = new Point(-1, -1);
}
```

9. Now for the drawing code. First we check to see if the LastPoint variable has been set. Then we check to see if a single finger is placed on the DiamondTouch, if it is we draw a line between the current point and the last point using the user specified colour. If it is not a single finger we erase the bounding region of the current touch.

```
private void GestureMove(GestureEvent gesture, TouchEvent e)
{
   if (!(LastPoint[e.ID].X==-1 && LastPoint[e.ID].Y == -1))
   {
      Graphics g = this.CreateGraphics();
      if (GestureIndicies.OneFinger == gesture.Index)
      {
         Pen userPen= new Pen(UserColors[e.ID]);
         userPen.Width = 5;
         g.DrawLine(userPen, e.Location, LastPoint[e.ID]);
      }
      else //erase drawings
         g.FillRectangle(BackColor, e.TouchArea);
    }
    LastPoint[e.ID] = e.Location;
}
```

10. At this point, we have completed our simple multimodal sketching example. You can run the project by pressing F5. It should look something like Figure 5.11. Press Alt-F4 to exit.

5.6 Discussion

The value of programming by demonstration. In an early version of GSI DEMO that did not include programming by demonstration I hand coded custom multimodal wrappers around several single user applications (e.g., Google Earth, Warcraft III, and The Sims) this process was laborious often taking weeks to create, with the systems being difficult to maintain and alter. After programming by demonstration was included in GSI DEMO, I redid these three wrappers as an experiment using programming by demonstration. All three application mappings took under an hour to do. Clearly the time savings were significant, as these mappings required an order of magnitude less time to complete. Programming by demonstration meant I could try out gestures, see how they worked, and discard them for other alternatives if desired.

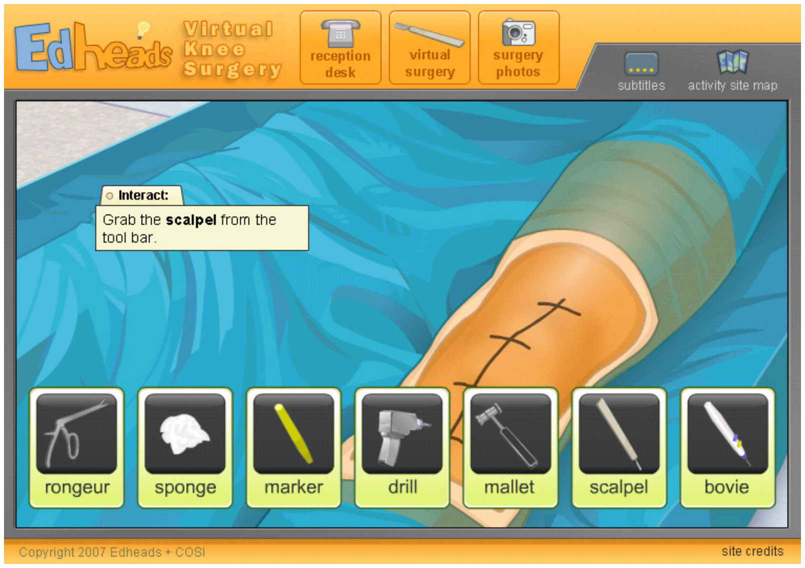


Figure 5.13 Virtual Knee Surgery Application from EdHeads.org

Customizing software to a multi user tabletop setting. I thought about the behavioural foundations of using speech and gestures in a multimodal setting (as described in Chapter 3). I adapted Virtual Knee Surgery, an educational tutorial produced by EdHeads.org of the steps involved with a knee surgery operation (Figure 5.13) for a multi user multimodal setting. This tutorial puts people in the role of a surgeon who must select the appropriate tools to perform the operation. By default, virtual surgery displays a list of tools on the bottom of the screen (closest to the upright user on the table). The user must click on one of the tools in order to select it. While this metaphor works well for a single user over an upright display, the fact that the buttons disappear once a selection has been made makes the action private and unapparent to other members around the table. While this metaphor works well for a single user over an upright display, it is hard to use in a multi user environment as buttons disappear, meaning that others cannot double check one's actions and provide assistance when necessary just like they would in a real surgery. I considered how the Virtual Knee Surgery application could be redesigned to fit this group setting. To facilitate double checking and assistance, I implemented tool selection as a set of verbal alouds in Virtual Surgery, and actions as a series of gestures. Just as a doctor would ask a nurse for a scalpel, participants can ask the computer to provide them with the appropriate tool and then do the action via an understandable gesture. This verbal aloud serves a double function as a computer command and as an awareness tool to inform collaborators about the actions that they are about to take. Different collaborators could also try out different surgical actions through gestures.

Single user limitations. When GSI DEMO is used to wrap existing single user applications it is fundamentally limited by the single user applications it wraps. While some multi-user interaction can be performed by interleaving individual actions, the underlying application is oblivious to this fact. As well, the mechanics of invoking certain application actions have side-effects, e.g., if a speech command invokes a graphical menu selection, the menu may briefly flash on the display. Suitable mappings can be difficult to create if the underlying application does not provide appropriate functionality. For example, it is not possible to provide a continuous panning gesture for a web document if only next page and previous page buttons are provided. Some applications may not be suited for multi user interaction on a tabletop display, e.g. it would be awkward for multiple people to simultaneously interact with text editors or programs that regularly open dialog boxes requiring sequential user interaction.

Programming by demonstration limitations. GSI DEMO is a crude programming by demonstration system. It makes many simplifying assumptions on how gestures should be mapped to application actions, and makes no attempt to learn across examples as done in other programming by example systems [Cypher, 1993] and even in other gesture recognition systems [Cao *et al.*, 2005]. For example, it assumes that there will be a simple mapping between a gesture (e.g., a whole hand move) and a mouse click/drag. While this makes it easy to map a hand move to a mouse panning action, mapping a two finger touch to a double tap is not possible. Similarly, complex gestural shapes (e.g., drawing a flower) are not recognized. Of course, an obvious step in this work is to incorporate techniques forwarded by the machine learning and programming by demonstration communities [Cypher, 1993, Cao *et al.*, 2005].

Multimodal mapping limitations. Only a simple speech + one gesture (e.g., finger, hand) can be mapped to a series of keyboard commands followed by a mouse click. Mouse click coordinates are remapped to the center position of the gesture. This means that complicated multimodal commands that involves clicking a menu before specifying a location would fail. I specifically chose not to support gesture movement mappings to keep GSI DEMO simple for end users. Thus end users need only worry about using the right hand posture to create a gesture mapping. Adding gesture movements would certainly increase the number of different possible mappings to existing single user applications. Macros and mappings can be deleted and reconstructed, but not edited. This means that long sequences would have to be recreated, even though a small edit would fix it. Again, methods exist to edit macros, e.g., Halbert's early SmallStar system [Cypher, 1993].

In spite of these technical limitations, I was able to define a reasonable working set of rich, natural actions atop our single user applications. Perhaps this is because the actions people want to do when collaborating over a surface are simple and obvious. Addressing some of the limitations described above will provide even more mapping possibilities.

Finally, GSI DEMO has not undergone formal evaluation. These are needed to both discover interface and conceptual weaknesses and validate basic ideas and workings.

5.7 CONCLUSION

In this chapter I described GSI DEMO, a toolkit designed to let people explore multi user multimodal speech and gesture interaction on a digital table. GSI DEMO allows collaboration-aware multi user multimodal applications to be built by programmers by exposing a simple Application Programmers' Interface. For those wishing to leverage existing off the shelf applications, GSI DEMO circumvents the tedious work needed to build collaboration-transparent speech and gesture wrappers from scratch. Training the computer is as easy as saying "Computer, when I do *[gesture action]* you do *[mouse action]*" or "Computer, when I say <*speech action*> you do *[mouse and keyboard macro]*". This gesture and speech infrastructure allows end users to focus on the design of their wrappers rather than underlying plumbing.

While GSI DEMO does have limitations (e.g., programming by demonstration, multimodal command construction), I believe that these limitations could be minimized by apply advanced programming by demonstration techniques and addressing its current limitations. However, I stress that my focus is not on doing programming by demonstration, but rather exploring multimodal co-located interaction and how these systems are used in practice.

Chapter 6. Multimodal Target Acquisition on Display Walls

***Objective Five**. I will compare the performance of using speech filtering for selection on a large digital wall display to two commonly used gesture-only selection techniques.*

In this chapter, I explore the use of speech and gesture commands for target acquisition on a large wall sized display. I examine the costs and benefits associated with using speech in tandem with gestures for distant freehand selection and compare its performance to common gesture based approaches. If the cost is too high in a single user setting, it will undoubtedly be too high in the multi-user setting.[4]

In this chapter, I present the *speech-filtered bubble ray* that uses speech to transform a dense target space into a sparse one. This technique builds on what people already do: people pointing to distant objects in a physical workspace typically disambiguate their choice through speech. For example, a person could point to a stack of books and say "the green one". Gesture indicates the approximate location for the search, and speech 'filters' unrelated books from the search. My technique works the same way; a person specifies a property of the desired object, and only the location of objects matching that property trigger the bubble size. I performed a controlled evaluation of using speech filtering for selection for selection on a large digital wall display. The results showed that people were faster and preferred using the speech-filtered bubble ray over the standard bubble ray and ray casting approach. While this study examines target acquisition performance for a single individual, selection is a basic activity that is heavily used in the multi user setting as well.

[4] Portions of this Chapter are published as:

Tse, E., Hancock, M. and Greenberg, S. (2007) Speech-Filtered Bubble Ray: Improving Target Acquisition on Display Walls. Proceedings of ICMI 2007, 307-314.

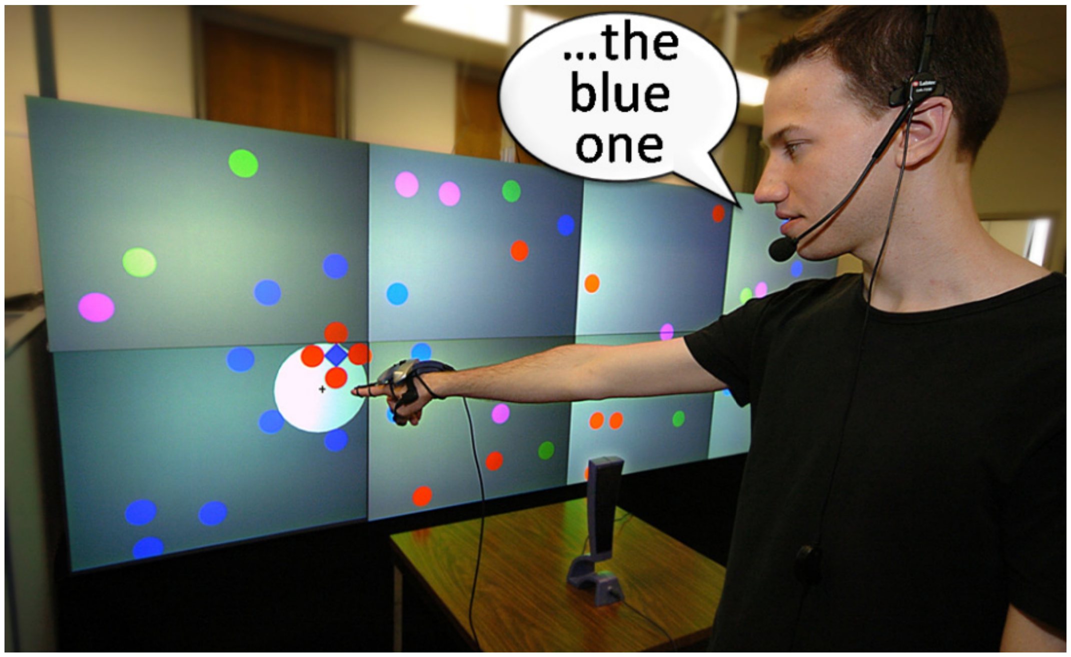


Figure 6.1 The wall study hardware configuration

6.1 Introduction

The recent development of large high-resolution wall display technology has been accompanied by parallel developments of suitable interaction techniques. Most systems still use input controls that are separate (vs. direct touch) from the wall, e.g., multiple mice [Tse and Greenberg, 2004b], game consoles, or remote controls that require a person to navigate through items via buttons. Touch-sensitive surfaces work well when people can reach the wall, e.g., Smart Technologies Interactive Whiteboards (smarttech.com) and the MERL DiamondTouch digital table [Dietz and Leigh, 2001]. Yet in many large display settings, some tasks are best performed from a distance. For example, in everyday conversations people may use nearby display walls to view and interact with content relevant to their discussion. They may find it inconvenient or interruptive to approach the display or acquire specialized input devices. If the display wall is large, some regions of the screen may not be easy to reach, e.g., the upper region may be out of reach for some people. The display wall's position within a furnished environment also impacts a person's proximity to it, e.g., when mounted out of direct reach in a public place like an airport or restaurant, or as a common information wall situated in a control room where operators are seated at workstations.

The general problem is: how can people effectively select items with large displays from a distance? As we will see in §6.2, a variety of strategies have been developed by others. Some move distant items closer to where the person is actually working on the display screen [Baudisch *et al.*, 2003, Bezerianos *et al.*, 2005]. Most others use variants of ray casting, where a person's pointing action is interpreted as a 'ray' hitting the screen. Ray casting is of particular interest as it is the most natural for people to do, and it also serves as a gesture that is easily understood by others involved in the activity.

While ray casting is reasonable for large targets, it is slow and error prone when people try to select small targets on the display. Indeed, I believe target acquisition will become a serious issue both as distances between people and the display increase, and as improved screen and input resolutions create more available pixels per inch. Fitts Law partially predicts this problem (see §6.2), but the situation is exacerbated by the natural shake in people's hands when pointing [Myers *et al.*, 2002], and by inaccuracies in ray casting input technologies. Even a very small shake when pointing translates to large jitters over a distance of several feet.

One way to get around this is by increasing the apparent size of small targets. In particular, Grossman *et al.* [2005] developed the bubble cursor to simplify pointing in a sparse environment; the cursor is surrounded by a 'bubble' resized to envelope the closest target. This technique works well if the surrounding space is fairly sparse since the bubble can grow reasonably large, but is ineffective in dense spaces as the bubble has little room to grow.

Thus the specific problem addressed by this chapter is: can we improve target acquisition via ray casting, even when targets are densely packed together?

I was inspired by observations of everyday communication: people often roughly point to an area containing several objects, and then use speech to discriminate the particular object of interest within that zone. For example, one might point to a coat rack containing several coats and ask "please, pass the red one". In this case, speech helps the listener filter out coats that are not red from the range of possible targets being pointing to by the speaker.

We claim that an analogous multimodal speech and pointing system, which we call speech-filtered bubble ray (or speech bubble for short), can help people select targets on a large interactive digital wall from several feet away, as illustrated in Figure 6.1. A person specifies a property of the desired object, and only the location of objects matching that property triggers the bubble ray size.

6.2 Related Work

There are two areas of relevant related work: input techniques that improve target acquisition by 'optimizing' Fitts Law parameters, and input techniques for distant freehand pointing.

6.2.1 Optimizing Fitts Law Parameters

Fitts Law is commonly used to model target acquisition [MacKenzie, 1989]. The Shannon formulation of Fitts Law [Mackenzie, 1995] states that the movement time (*MT*) that it takes to acquire a target of width *W* and distance (or amplitude) *D* is predicted by:

$$MT = a + b\log_2\left(\frac{D}{W} + 1\right)$$

a and *b* are empirically determined constants, and the logarithmic term is called the ***index of difficulty*** (*ID*). The equation predicts that smaller target widths and larger distances (from the current location) will increase selection time. Thus target selection can be improved by decreasing *D*, by increasing *W,* or both.

Decreasing Target Distance (*D*). Baudisch *et al.* [2003] reduce target distance by bringing distant targets closer to the user. Their *Drag-and-Pop* method analyzes the directional movements of the cursor, and then brings virtual proxies of the potential targets towards the cursor (e.g., a folder or application). Studies of drag-and-pop showed selection to be faster for large target distances. However, their method cannot determine when the user intended to select distant targets versus those nearby. Thus the presence of distant objects can make selection difficult for nearby targets.

Bezerianos *et al.*'s *Vacuum* method [2005] is somewhat similar, but also allows the user to control the angle of distant targets that they were interested in and supports multiple object selection. Selection time was found to be similar for single targets but significantly faster for multiple target selection.

Increasing Target Width (*W*). Kabbash and Buxton [1995] increased the target width by increasing the cursor size. Instead of a single pixel hotspot as seen in standard cursors, area cursors have a larger active region for selection. By setting *W* to be the width of the area cursor, they showed that selection of a single pixel target could be accurately modeled using Fitts Law. Thus, very small targets are easier to acquire. However, area cursors are problematic in dense target spaces where multiple targets could be contained in an area cursor.

McGuffin and Balakrishnan [2005] increased the target size dynamically as the cursor approached. They found that users were able to benefit from the larger target width even when expansion occurred after 90% of the distance to the target was traveled. They also showed that overall performance could be measured with Fitts Law by setting the width to the size of the expanding target.

Increasing *W* and Decreasing *D*. A different approach dynamically adjusts the control-display gain (*C:D*). By increasing the gain (cursor speed) when approaching a target and decreasing it while inside a target the motor space distance and width are decreased and increased, respectively. Blanch *et al.* [2004] showed that performance could be modeled using Fitts Law, based on the resulting larger *W* and smaller *D* in motor space. However, problems arise when there are multiple targets, as each slows down the cursor as one travels towards it.

As mentioned, Grossman *et al.* developed the Bubble Cursor to ease target acquisition in a sparse display [2005]. The cursor is surrounded by a dynamically resizing bubble so that only the closest target is enveloped by the bubble. An example is shown in Figure 6.2 (left): the bubble around the cross hair cursor expands until it just touches the nearest target. This effectively increases target width (since the bubble gets bigger), and decreases target distance (because less distance needs to be traveled to reach the target). The problem is that other nearby targets, called *distracters*, limit the size of the bubble. For example, if the four objects surrounding the cursor in Figure 6.2 (left) were closer together, the bubble would be much smaller. In other words, the width of the target is dependent on the distance of the closest distracters adjacent to it, as it expands so that only the closest target is selected at any time. This new target size is called the *Effective Width* (*EW*). Their study shows that Bubble Cursor's performance can be modeled using Fitts Law by setting $W = EW$.

6.3 Freehand Pointing at Large Displays

Ray Casting is a commonly used technique for pointing to distant objects on a large display (e.g., [Bolt, 1980, Vogel, 2005]), where the cursor is drawn as the intersection of the ray from the hand/pointer and the screen. Laser pointers are an obvious candidate for implementing ray casting, and many people have explored how they can be implemented and used. For example, Myers *et al.* [2002] considered different laser pointer form factors (a pen, a laser pointer mounted on a glove, a scanner, and a toy gun) to see how they minimized hand jitter and affected aiming. Parker *et al.*'s *TractorBeam* [Parker *et al.*, 2005] affords selection on a tabletop display by having people point the tip of the six degree-of-freedom pen at distant targets. Other ray casting devices include data gloves, wands tracked by motion capture systems, and so on.

Improving Ray Casting. The basic selection methods described in §6.2.1 can be applied to ray casting. For example, the *TractorBeam* [Parker *et al.*, 2005] includes: Expand Cursor (cursor expands when close to the target), Expand Target (target expands when cursor approaches) and Snap-to-Target (cursor jumps to the closest target). The Snap-to-Target proved quickest but had a high error rate. Expand Cursor was slowest and proved problematic when multiple targets were nearby, but had the lowest error rate.

Selection. While target acquisition is all about pointing, a complementary act is to indicate the actual act of selection. In a conventional computer, the mouse may move the cursor, but it is the button-down operation that 'selects' the object under the cursor. In distant pointing, a common approach is to use a selection button or similar control onto the pointing device, e.g., [Myers *et al.*, 2002]. Vogel and Balakrishnan [2005] present two gesture-based selection methods applied to distant freehand pointing for large high-resolution displays. With the *thumb trigger* the user presses their thumb against their hand to perform a selection. With the *air tap*, the user moves their index finger to indicate selection.

Multimodal input. Early attempts at distant freehand pointing include Bolt's [1980] Put-That-There multimodal system where individuals could interact with a large display via speech commands qualified by deictic reference, e.g., "Put that…" (points to item) "there…" (points to location). Bolt argues and Oviatt confirms [Bolt, 1980, Oviatt, 1997] that this multimodal input provides individuals with a briefer, syntactically simpler and more fluent means of input than speech alone. These systems, however, still use basic ray casting for target acquisition, while the speech channel directs the system on what to do with the selected target. For example, "Put that…" is analogous to a mouse-down event selecting the target pointed at, and "there" is like the mouse-up release specifying that the new target is the final location.

Multimodal selection in VR environments. Since Bolt's pioneering system, several researchers have explored multimodal interaction with 3D environments. Kaiser *et al.* [2003] implemented multimodal selection for an augmented reality interior decoration environment. This system fused the n-best recognition results of speech, gesture, and gaze input to disambiguate the object that a person was pointing to. Studies revealed that this approach was effective for handling recognition errors and uncertainty in the recognition of any one modality. However, their study did not provide visual feedback of the target search space as is provided by the bubble cursor, also they did not compare this technique against the unimodal (i.e. gesture only) selection techniques typically used to interact with large wall displays.

6.4 Speech-Filtered Bubble Ray

Many of the techniques described in §6.2, whether for direct touch, separate input controls, or ray casting variants, attempt to 'optimize' Fitts Law performance by decreasing target distance or increasing target width. Most only work well in a sparse space as they use the empty areas to infer ways to increase effective width or decrease effective distance. In a dense space, nearby distracters (i.e., other potential targets) limit their effectiveness; in the worst case, they degrade to simple ray casting. Yet in practice, dense information spaces are more typical than sparse ones, e.g., an icon-filled desktop, a web page or document filled with text, a control panel comprising many widgets (buttons, sliders, etc.), or a highly populated node and link graph. The difficulty of target acquisition via free-hand pointing is particularly onerous at a distance due to pointing inaccuracies of the input device and hand shake. The question is: how can we improve target acquisition in such a densely packed space?

As introduced earlier, people already use speech to qualify pointing gestures made with one's hands. Mary might point to a region on a book shelf with her finger and say "the green one", or Fred might point to a file browser on a large display and say "the Latex file" (Figure 6.1). Viewers interpret the gesture as indicating the rough regions they should be attending, and then use the speech act to decide upon the object of interest. Both speech and gesture work in concert; neither provides enough information by itself to discriminate the object of interest.

Our new interaction method works on the same principle. First, I adapted the bubble cursor to work with freehand ray-casting (*vs.* a mouse) for distant selection – I call this the ***bubble ray***, but it is otherwise identical to the bubble cursor. Second, I added speech-filtering capabilities to it to create the ***speech-filtered bubble ray***. As a person moves the bubble ray towards an object, he or she concurrently uses speech to inform the system of a particular property of that object. At that point, objects that don't have that property are filtered from consideration by the bubble cursor algorithm. Unless objects with the same property are very close to one another, the effect is that speech filtering makes densely packed target spaces sparser. This approach is similar to Sense Shapes [Kaiser *et al.*, 2003], the system projects 3D cones from a person's hand onto the projection surface, yet with speech bubble ray the size of the cone dynamically expands and contracts so that only one 2D target is selected on wall display.

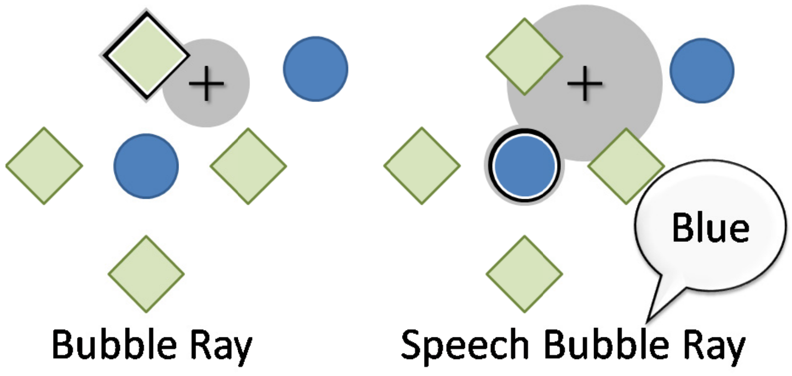


Figure 6.2 Bubble Cursor and Speech Bubble Ray methods

To illustrate this by example, consider Figure 6.2, which shows a standard bubble ray implemented as a bubble ray on the left and our speech-filtered bubble ray on the right. Say the person wants to select the blue circular object in the middle, and she is moving the cursor towards it from the top. If the person does not say anything, the system behaves as a normal bubble ray, where the bubble expands to touch and select the closest target. As Figure 6.2 (left) shows, the bubble is somewhat small because other distracter targets are nearby. However if the person says "the blue one", the bubble ray selection space is filtered to include only blue objects and to ignore the green diamonds, and the bubble ray expands accordingly (Figure 6.2 right, also Figure 6.1). Here, the bubble's size is constrained by the next closest blue object. Comparing the two figures illustrates that even though the blue objects are not that far apart, the bubble is considerably larger and thus target acquisition is much easier for the speech bubble. Selection can then be performed with techniques such as a button press on a control held in the non-dominant hand, or through a gesture [Vogel *et al.*, 2005]. By simply saying "none" or "cancel" the user can remove the speech filter.

For the best effect, the speech qualifier should be said early in the target acquisition process, e.g., before or during the ballistic move just as one visually tracks the object and recognizes one of its properties. However, we stress that the speech qualifier does not perform any selection. We chose this design as there may be many objects near the target that share a similar property, and an incorrect selection could be made if speech filtering also performed a selection.

By filtering according to speech commands, our technique increases the effective width of targets. Fitts Law predicts that this increase will reduce movement times [Fitts, 1954]. However, the technique has an added cognitive overhead of deciding on what to filter and an added time to actually speak a command. Consequently, we performed an empirical study that compares the performance cost of our speech-filtered bubble ray to two other distant freehand pointing techniques. These include ray casting (commonly used as a control in such studies) and the bubble ray.

6.5 User Study Method

Our study goal was to compare selection times and error rates between three pointing techniques: ray casting, bubble ray and speech-filtered bubble ray.

6.5.1 Participants

Thirty students (17 male, 13 female) from a local university participated in the study. Ages ranged from 18-34; all participants were right-handed and daily computer users with normal or corrected to normal vision. Half of all participants (15) had experience with large display pointing devices (e.g. Smart Board, Nintendo Wii). When asked to rate their English fluency on a scale of 1 to 5 (5 being completely fluent), 27 participants rated themselves as 5, one rated herself as 4, one rated himself as 3, and one rated himself as 2. When asked about what language they primarily spoke at home, 21 reported English with the remaining 9 reporting other languages – Asian, Middle Eastern and French.

6.5.2 Freehand Pointing and Selection Techniques

We adapted three selection techniques for use in our environment.

Ray casting is implemented as a crosshair cursor that represents the intersection of a ray starting from a person's hand and intersecting with the display wall. As the person moves their hand, the corresponding crosshair also moves.

Bubble ray adapts Grossman *et al.*'s bubble cursor technique [2005] for distant freehand pointing on a wall display. Our version differs only in that the person uses ray casting instead of a mouse cursor. The bubble around the crosshair dynamically expands so that only the nearest target is enveloped in the bubble (Figure 6.2, left). We use the formula described in [Grossman *et al.*, 2005] to control the bubble's size. We note that bubble cursor as applied to ray casting has not been evaluated elsewhere, but we expect that the performance can be modeled (as it is with the mouse in [Grossman *et al.*, 2005]) using Fitts Law by setting the target width (W) to the effective width.

Speech-filtered bubble ray (or **speech bubble** for short) is visually identical to bubble ray for the purposes of the experiment, except that the bubble size is adjusted to the filtered objects. This is actually a 'worst case' visualization, as in practice filtered targets could be faded to emphasize the sparseness of the new selection space. Using a microphone headset, people spoke a single property of the target (its color) into a speech recognizer to activate the filter.

For all three techniques, if the crosshair or bubble is within a target's active region, the target is highlighted with a white and black border (see Figure 6.2). This emphasis is visually similar to the underlining of links on a web page or the blue highlight seen with the single-click icon selection mode in Microsoft Windows XP. This is especially important for both bubble ray approaches, as it emphasizes that a target has been acquired and that there is no need to further move the cursor closer to the target.

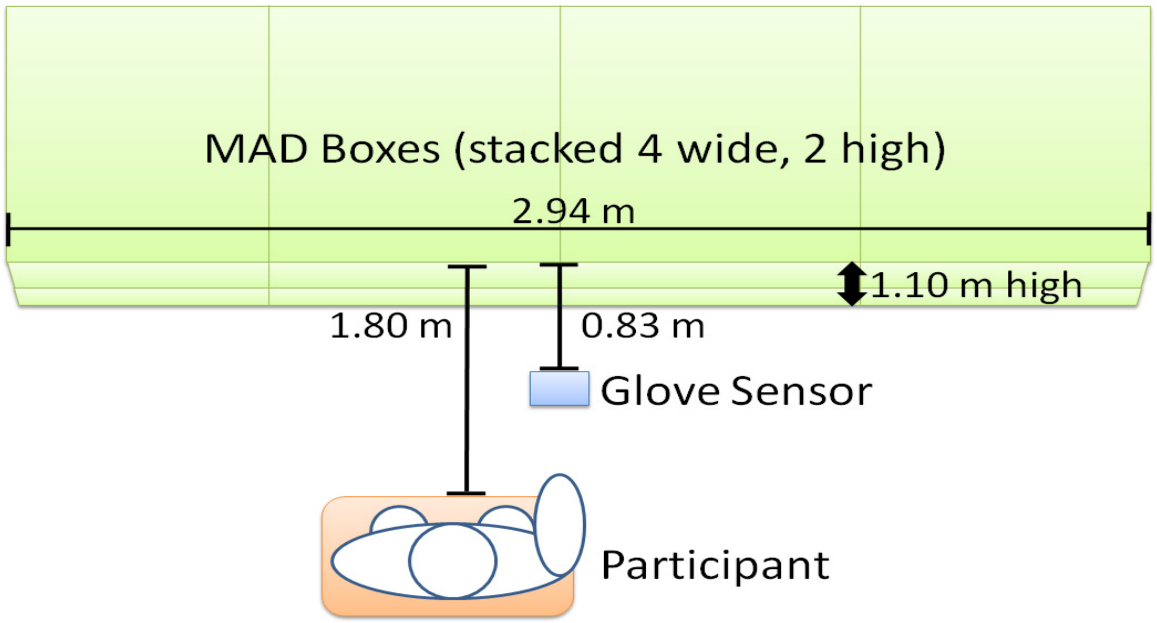


Figure 6.3 Birds-eye-view of our experimental layout

6.5.3 Apparatus

As seen in Figure 6.1 and Figure 6.3, we used a 2.94 m x 1.10 m display surface composed of eight modular ambient display (MAD) boxes [Schmidt *et al.*, 2004] each containing a 1024 x 768 LCD projector stacked four high and two wide for a total resolution of 4096 x 1536. All projectors were connected to a single workstation with two Matrox QID Pro display adapters that each support four displays. Our system is designed in C++ using a large OpenGL window spanning across eight displays.

For input we used a six degree-of-freedom Essential Reality P5 Data Glove, a low cost input device intended for computer gaming. We used only the *x* and *y* values for our experiment, thus the position of the cursor was only affected by the position of the glove relative to the sensor. Tilting the hand would not change the position of the cursor.

As seen in Figure 6.3, the data glove sensor was placed at the bottom-centre and 0.83 m in front of the wall. Participants were asked to stand in a square marked by masking tape 1.80 m in front of the wall, shifted 0.83 m to the left of centre, so that the right arm of the participant was aligned at the centre of the screen. Freehand pointing was performed with the right arm.

Participants used a Labtec LVA 7330 noise-cancelling microphone for the speech bubble technique. Because we did not want speech recognition errors to influence our results, we used a Wizard of Oz speech recognition technique: the target colour was activated when any speech was recognized. If the participant said the wrong target colour, the experimenter would mark the trial as having a speech error.

We gave participants a wireless slide remote to perform selections in the non-dominant hand. We preferred this to a selection technique in the dominant hand to minimize any drift from the intended selection location.

As a side note, we expect that future vision and audio processing systems can easily detect user actions without the need for specialized glove tracking devices and headsets.

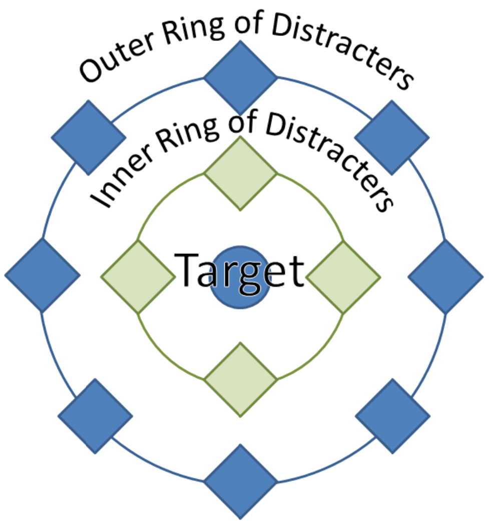


Figure 6.4 Distracter layout

6.5.4 Task

For each trial, participants were presented with a screen full of targets coloured red, green, blue, and pink, visible in Figure 6.1 and somewhat similar to a tiling of the pattern seen in Figure 6.4. One target was presented in a different shape than the rest (either a diamond or a circle). Participants were asked to select the differently-shaped target as quickly and as

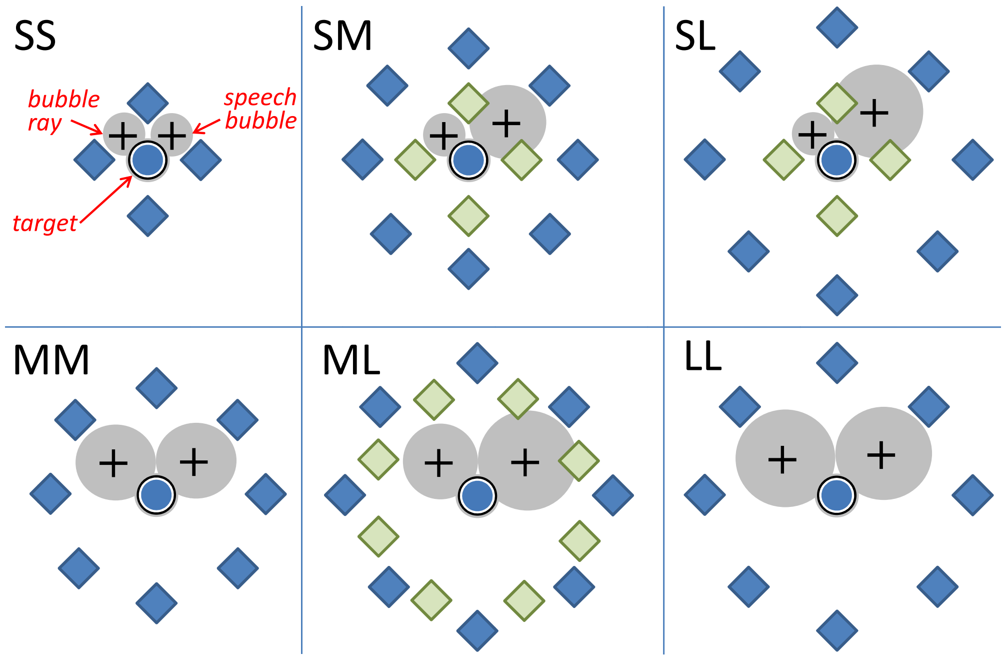


Figure 6.5 The six distracter layout conditions

accurately as possible.

6.5.5 Design and Procedure

We used a repeated measures within-participant factorial design. Our independent variables were:

- *technique* (ray casting, bubble ray, speech filtering)
- 6 *distracter layouts* as configured in different inner and outer ring widths, and which affect how large the bubble ray and speech bubble ray can grow (Figure 6.4 and Figure 6.5).

Table 6.1. The six distracter layouts

Acronym	**Inner Ring (different colour from target)**	**Outer Ring (same colour as target)**
SS	None	Small (6.5 cm from target centre)
SM	Small (6.5 cm)	Medium (14.1 cm)
SL	Small (6.5 cm)	Large (24.7 cm)
MM	None	Medium (14.1 cm)
ML	Medium (14.1 cm)	Large (24.7 cm)
LL	None	Large (24.7 cm)

The six distracter layouts are a combination of two factors: inner width and outer width (Figure 6.4, Table 6.1). These two factors could not be considered separately because the outer width was constrained to be no smaller than the inner width (and thus, they are not independent). *Inner ring* consists of distracters coloured differently than the target, thus its distance from target restricts bubble size only in the bubble ray condition. *Outer ring* consists of distracters of the same colour as the target, thus its distance restricts the bubble size in the speech bubble condition once the speech command has been spoken. For example, in the small inner / large outer ring width condition (Figure 6.5, top-right), the bubble ray is constrained by the small inner ring of distracters (left bubble) and the speech bubble is constrained by the large outer ring of distracters (right bubble).

Both the inner and outer widths vary from small (6.5 cm from target centre), to medium (14.1 cm), and large (24.7 cm). The small size is typical of targets stacked side-by-side (e.g., lines in a text document), medium is similar to the separation of file icons in a folder, and the large size represents a sparse space on a desktop.

Figure 6.5 and Table 6.1 shows the six combinations of inner and outer widths. For each condition, the minimum size of the bubble ray is shown on the left, and the minimum size of the speech bubble (once a colour has been spoken) is shown on the right. For the conditions where the inner and outer widths are the same (top-left, bottom-left, and bottom-right), only one ring of distracters of the same colour as the target need be shown (as this ring is sufficient to limit the size of the bubble in both the bubble ray and speech bubble conditions). We will refer to these six conditions as: SS, SM, SL, MM, ML, and LL.

We kept constant the distance to the next target (87.5 cm), the diameter of the targets (6.4 cm), the number of non-overlapping distracter targets placed randomly around the screen (74), and the number of possible target colours (4: red, green, blue, pink). All targets had a circular activation area regardless of the shape shown on the screen (diamond or circle).

Participants completed each technique and the distracter layout combinations six times, for a minimum of 108 trials per participant. If a participant made an error during a trial, either by selecting the wrong target or saying the wrong speech command, the trial was repeated. A brief sound cue would indicate if the correct or incorrect selection was made.

Presentation of the three techniques was counter-balanced using a Latin Square. The experiment consisted of three blocks (one per technique), with each block following the procedure of:

- 36 practice trials
- 36 trials
- Incorrect trials repeated
- Questionnaire (what did you like/dislike about the technique?)

The practice trials repeated exactly the same conditions seen in the experiment. Each block of 36 trials was randomized. Participants were asked to complete a post-test questionnaire asking them to compare each of the three techniques after the experiment.

6.5.6 Hypotheses

We had the following hypotheses for this experiment:

H1: The speed of selection will not vary for ray casting.

H2: Bubble ray will be faster in proportion to the inner ring width.

H3: Speech bubble will be faster in proportion to the outer ring width.

H4: Bubble ray will be faster and result in fewer errors than ray casting when the inner ring width is either medium or large.

H5: Speech bubble will be faster and result in fewer errors than ray casting when the outer ring width is either medium or large.

H6: Speech bubble will be faster and result in fewer errors than bubble ray when the outer width is larger than the inner width.

H7: Bubble ray will be faster than speech bubble when the inner ring width and outer ring width are the same.

We hypothesize H7 because of the added overhead in speech bubble of both determining and speaking the command.

6.5.7 Data Collection

During the experiment I logged the position of the cursor, the time, speech volume, and the closest target every 10 milliseconds. When a selection was made we recorded the total trial time, any selection or speech errors (marked by the experimenter) and recorded the positions and colours of every target on the screen.

6.6 Results and Discussion[5]

To analyse our data, we performed a 6 (distracter layout) × 3 (technique) within-participants ANOVA. We used both target selection time and number of errors (either incorrect spoken command or missed targets) as dependent measures. We performed the same two analyses with the additional between-participants factor of gender and found no additional main effects or interactions. We present the two-way ANOVA for simplicity.

6.6.1 Speed

There was a main effect of technique ($F(2,58) = 29.5$, $p < .001$). Post-hoc comparisons showed that all pairwise differences were significant ($p < .01$) and that participants selected targets the fastest with the speech bubble ($M = 2.62$ s, $SD = 0.09$ s) followed by the bubble ray ($M = 2.97$ s, $SD = 0.09$ s), and the ray casting technique was slowest ($M = 3.42$ s, $SD = 0.12$ s).

There was a main effect of distracter layout ($F(5,145) = 47.0$, $p < .001$). There was also significant interaction between distracter layout and technique ($F(10,290) = 10.1$, $p < .001$). While we expected the former main effect (changing target size should affect speed of target acquisition), we were most interested in how these changes affect each of the techniques differently. Thus, we will only discuss this latter interaction. We present pairwise differences broken down both by technique and by distracter layout as they are both illustrative.

Figure 6.6 shows the target selection times for each distracter layout separated by technique. Table 6.2 shows significant pairwise differences for distracter layout pair. These pairwise differences partially confirm hypotheses H1, H2, and H3. We found no significant differences in selection times for the ray casting technique (H1) with the exception of the SS condition being slower than the rest. This exception is likely due to the fact that, in the SS condition, the pattern of the single ring of targets is smaller as a whole than in any other condition, making it more difficult to recognize the target before acquiring it and thus

[5] To clarify, Mark Hancock performed the quantitative analysis.

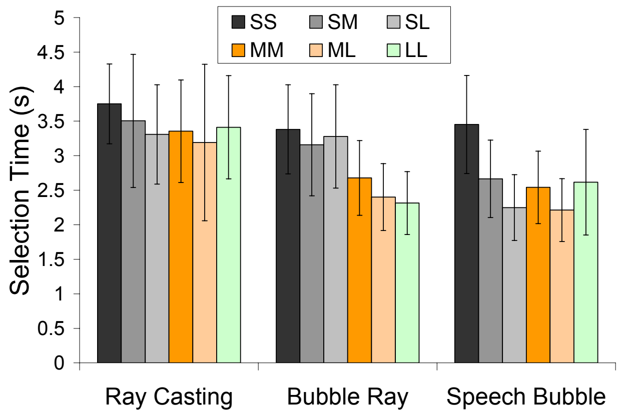


Figure 6.6 Target selection times for distracter layout, separated by technique.

Table 6.2. Distracter layout time differences separated by technique. Pairwise significance values are in bold.

Distracter Layout	Ray Casting	Bubble Ray	Speech Bubble
SS vs. SM	*p* = 0.11	***p* = 0.1**	***p* < .001**
SS vs. SL	***p* < .001**	*p* = 0.37	***p* < .001**
SS vs. MM	***p* < .01**	***p* < .001**	***p* < .001**
SS vs. ML	***p* < .01**	***p* < .001**	***p* < .001**
SS vs. LL	***p* < .01**	***p* < .001**	***p* < .001**
SM vs. SL	*p* = 0.20	*p* = 0.36	***p* < .001**
SM vs. MM	*p* = 0.35	***p* < .001**	*p* = 0.14
SM vs. ML	*p* = 0.06	***p* < .001**	***p* < .001**
SM vs. LL	*p* = 0.5	***p* < .001**	*p* = 0.65
SL vs. MM	*p* = 0.76	***p* < .001**	***p* < .001**
SL vs. ML	*p* = 0.50	***p* < .001**	*p* = 0.55
SL vs. LL	*p* = 0.35	***p* < .001**	***p* < .01**
MM vs. ML	*p* = 0.39	***p* < .001**	***p* < .001**
MM vs. LL	*p* = 0.65	***p* < .001**	*p* = 0.41
ML vs. LL	*p* = 0.18	*p* = 0.21	***p* < .001**

increasing cognitive load. In the bubble ray condition, the smallest inner width was slower to select than larger inner widths, confirming H2. The MM condition was also slower than the LL condition, as H2 predicts, however, the ML condition was unexpectedly faster than the MM condition. We suspect this exception is again due to the fact that, in the ML condition, the distracter layout of surrounding targets improved the participants' ability to recognize the location of the center target, artificially improving selection time for this condition. In the speech bubble condition, the targets with a small outer width were slowest to select (H3), the targets with a medium outer width were also slower to select than those with a large outer width (H3) with the exception of the LL condition. Again, the visual cues provided as a side effect of our setup may have been the cause of this exception. In the LL condition, there is only one ring of distracter targets, distant from the actual target, making the pattern of targets more difficult to recognize.

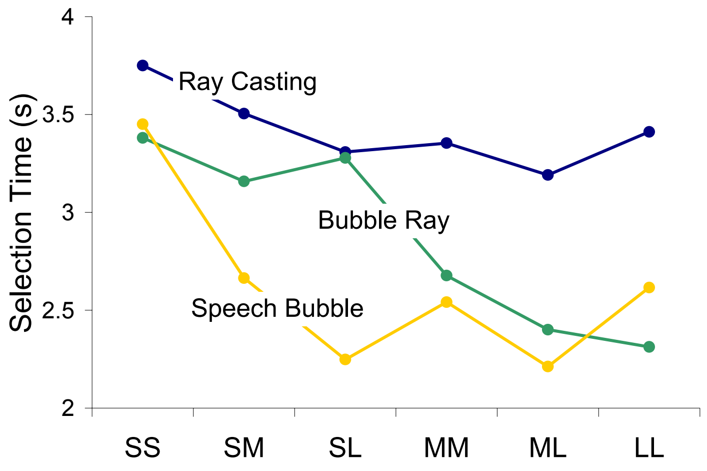


Figure 6.7 Target selection times for each technique, separated by distracter layout.

Table 6.3. Time Differences between techniques separated by distracter layout. Pairwise significance values are in bold.

Distracter Layout	Ray Casting vs. Bubble Ray	Ray Casting vs. Speech Bubble	Bubble Ray vs. Speech Bubble
SS	***p* = .02**	***p* = .04**	*p* = .60
SM	*p* = .08	***p* < .001**	***p* < .001**
SL	*p* = .83	***p* < .001**	***p* < .001**
MM	***p* < .001**	***p* < .001**	*p* = .26
ML	***p* < .001**	***p* < .001**	***p* = .02**
LL	***p* < .001**	***p* < .001**	***p* = .04**

Figure 6.6 shows the target selection times for each technique separated by distracter layout. Table 6.3 shows the significant pairwise differences for each. These pairwise differences confirm hypotheses H4, H5, and H6. As H4 predicts, the bubble ray technique was significantly faster than ray casting whenever the inner ring width was medium or large (MM, ML, LL). As H5 predicts, the speech bubble technique was significantly faster than ray casting whenever the outer ring width was medium or large (SM, SL, MM, ML, LL). As H6 predicts, the speech bubble technique was faster than the bubble ray technique whenever the outer ring width was larger than the inner ring width (SM, SL, ML). As H7 predicts, speech bubble was significantly slower than bubble ray in the LL condition, likely due to the overhead required in speaking the command. In addition to our predicted results, we found

that ray casting was significantly slower than both bubble ray and speech bubble in the SS condition.

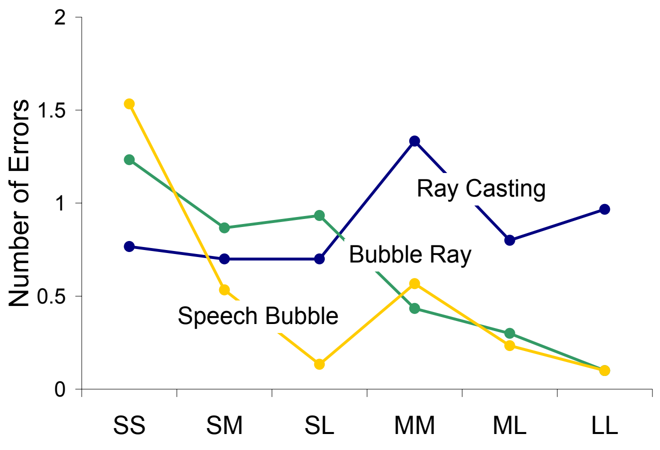


Figure 6.8. Number of errors for each technique, separated by distracter layout.

Table 6.4. Error differences between techniques, separated by distracter layout, pairwise significance values in bold.

Distracter Layout	Ray Casting vs. Bubble Ray	Ray Casting vs. Speech Bubble	Bubble Ray vs. Speech Bubble
SS	*p* = .13	*p* = .03	*p* = .44
SM	*p* = .55	*p* = .49	*p* = .13
SL	*p* = .26	***p* < .01**	***p* < .001**
MM	***p* < .01**	***p* < .01**	*p* = .40
ML	***p* =.03**	***p* < .01**	*p* = .54
LL	***p* < .001**	***p* < .001**	*p* = 1.00

6.6.2 Error

The average number of errors for any trial was 0.7 (*SD* = 1.0). Due to the small number of errors, not much can be read from these differences. However, some of these differences were statistically significant. There was a main effect of technique (*F*(2,58) = 5.7, *p* < .01). Post-hoc comparison revealed that participants performed significantly more errors with ray casting than with speech bubble (*p* < .01). There was no significant difference between bubble ray and either ray casting (*p* = .07) or speech bubble (*p* = .19). There was a main effect of distracter layout ($F(5,145) = 10.6$, $p < .001$) and an interaction between distracter layout and technique ($F(10,290) = 5.6$, $p < .001$). We will again only discuss the interaction.

Figure 6.8 shows the number of errors for each technique separated by distracter layout. Table 6.4 shows the significant pairwise differences for each. These pairwise differences further support our results for speed and confirm hypotheses H4, H5, and H6. For H4, the bubble ray resulted in significantly fewer errors than ray casting when the inner width was medium or large (MM, ML, LL). For H5, speech bubble resulted in significantly fewer errors than ray casting when the outer width was medium or large (SL, MM, ML, LL). This trend also existed for the SM condition, but was not significant. For H6, speech bubble resulted in significantly fewer errors than bubble ray when the difference between inner and outer widths was the largest (SL), but this difference was not significant for the SM or ML conditions.

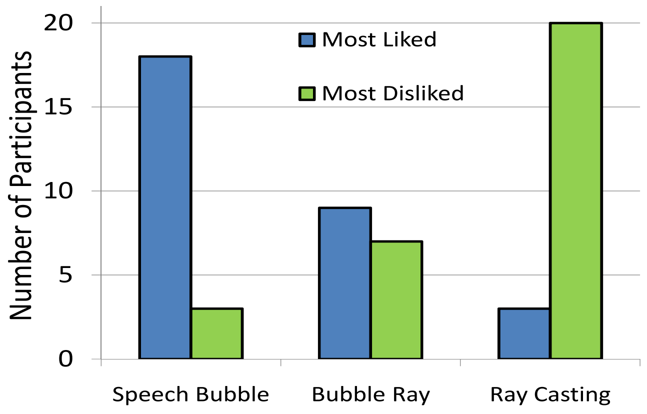


Figure 6.9 Participant responses of most liked and most disliked technique

6.6.3 Questionnaires

Participant's post-test questionnaire responses revealed a preference for speech bubble as the most liked and easiest technique to use.

Figure 6.9 shows participant responses to the most liked and most disliked method: 18 liked speech bubble the most, 9 chose bubble ray, and 3 chose ray casting. When asked about which technique they most disliked the opposite effect was observed: ray casting was chosen by 20 participants, 7 disliked bubble ray and 3 disliked speech bubble. Participants' comments reflected their selections as one participant wrote that speech bubble "...makes it easier to select the different shapes by filtering color" while ray casting "was the least forgiving". A few disliked the bubble ray technique saying "I didn't like how the bubble changes in size" and "the jittering of the size of the bubble became a distraction". This problem is caused by the natural shake in people's hands and is further exacerbated by the noise present in the input device.

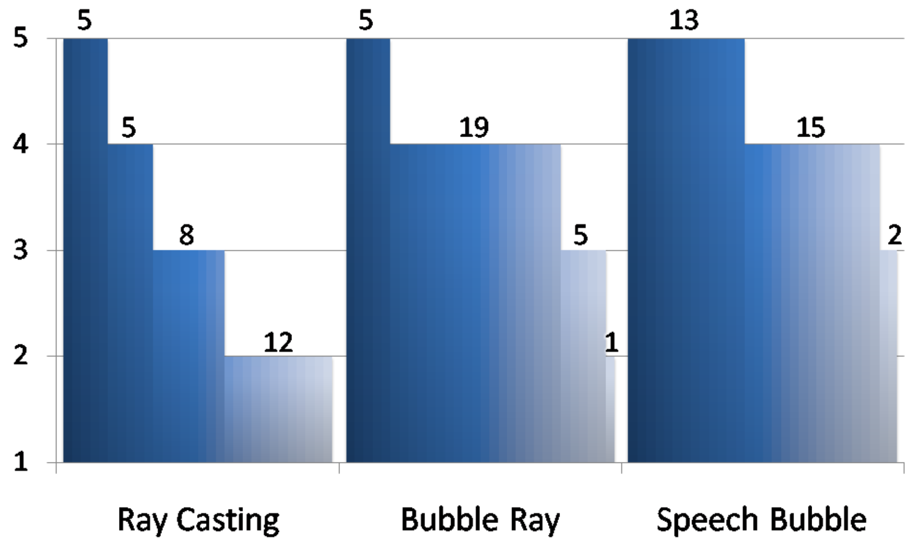


Figure 6.10. Individual participant breakdown to "I found technique easy to use", 5-strongly agree, 1-strongly disagree

Participants were also asked to say if they agreed that each technique was easy to use on a 5-point Likert scale (1 being strongly disagree, 5 being strongly agree) As seen by the area covered by each technique in Figure 6.10, speech bubble was ranked higher (*M* = 4.4, *SD* = 0.6) than bubble ray (*M* = 3.9, *SD* = 0.7) and ray casting (*M* = 3.1, *SD* = 1.1). For the speech bubble condition, all participants ranked its ease of use as either strongly agree, agree or netural. Ray casting had the highest percentage of neutral and disagree responses (20 of 30 participants).

For the people who did not favour the speech bubble, some stated that ray casting was easy to use because it was "like a mouse pointer" and thus most closely matched their everyday use of a computer mouse. Others preferred ray casting and bubble ray over the speech bubble because they "didn't have to speak". Some participants also found wearing a headset microphone uncomfortable.

Language and gender deserve mention. I examined the most liked preferences broken down by gender and language spoken at home but did not notice any notable effects. Some non-native English speakers commented that "coordination between speech and pointing of objects was a bit confusing / delaying".

6.6.4 Overall Discussion

These results show that the speech bubble technique provides the performance gain that we had expected and that speech bubble is preferred by most people. Specifically, all our hypotheses were confirmed, suggesting that speech filtering can benefit target selection by effectively increasing target width, even for densely-packed targets. In particular, the speech bubble technique performed as well or better than ray casting and bubble ray in most cases. The only exception was the large inner / large outer width condition, suggesting that the added overhead of speaking the command becomes slightly detrimental when the surrounding targets are very sparse and speech filtering provides no expected benefit. However, this degradation is only to a level slightly less than bubble ray, but is still faster and less error prone than ray casting. In practice, a person might simply choose not to speak the property, and performance would resort to using bubble ray (the default behaviour when no filter is spoken). In all other conditions, the overhead of speaking was negligible, and was far outweighed by the benefit of filtering.

I also mentioned several other techniques in §6.2 whose performance is compromised by nearby distracters [Baudisch *et al.*, 2003, Bezerianos *et al.*, 2005, Blanch *et al.*, 2004, Kabbash *et al.*, 1995, McGuffin *et al.*, 2005]. I believe that speech filtering could be applied to these techniques as well, with performance gains similar to our speech bubble.

6.7 Design Challenges

A challenge in designing interactions that leverage speech-filtered bubble ray is that the system must reveal properties of the targets that need to be selected. Of course, for bubble ray, the position of every target must be revealed. For speech bubble, the properties of the target used for filtering must also be revealed. For example, in a large desktop environment, the system would need to reveal/detect the position, colour and names of targets if those properties were to be used for speech filtering.

As with bubble cursor, the visual distraction of the bubble can hinder performance when there are very few targets and the bubble grows to become larger than the size of the screen. This problem is exacerbated when speech filtering is used to filter dense target spaces. To correct for this problem, a size limit can be placed on the bubble to limit the visual distraction. Alternatively, a gradient could be used to fade the bubble to transparent past a fixed size, so that the bubble can be much larger with less visual distraction.

6.8 Conclusion

In this chapter, I introduced the *speech-filtered bubble ray*, a technique for improving target acquisition using distant freehand pointing on large display walls by using properties of the target to filter the selection space. While Fitts Law suggests that performance of this method is better than a standard bubble ray or ray casting, speech filtering does incur some cost as people need to determine the target property to filter and say it out loud. I compared the performance of speech filtering to two commonly used gesture-only selection techniques. The empirical results provide evidence that the benefits of speech filtering (even when additional visual effects are omitted) significantly outweigh these costs, and effectively make dense target spaces sparser.

This chapter extends the motivations (established in Chapters 3 and 4) for using speech and gesture commands in a collaborative setting. In addition to the dual purpose nature multimodal commands as commands to the computer and communication to others, this chapter demonstrates that multimodal commands can be designed to improve selection efficiency on a large display.

Chapter 7. Multimodal Split View Tabletop Interaction Over Existing Applications

***Objective Six**. I will design and implement a collaboration-transparent multimodal split view table to support parallel work over existing applications.*

As explored in previous chapters, multi user multimodal interaction over a digital table with collaboration-transparent off the shelf applications is limited by the one user per computer assumption of current operating systems. That is, people cannot work in parallel as the system can only support a single stream of keyboard and mouse input. This chapter pushes the boundaries of using existing single user applications by using two computers over a shared multimodal digital table.[6]

In this Chapter I present multimodal split view interaction: a tabletop whose surface is split into two adjacent projected views. If each person works on a separate computer they can effectively work in parallel. There are three ways that existing applications can be leveraged on a split view tabletop. Independent applications let people see and work on separate systems. Shared screens let people see a twinned view of a single user application. Collaboration-aware groupware lets people work in parallel over large digital workspaces. Atop these, I add multimodal speech and gesture interaction capability to enhance interpersonal awareness during loosely coupled work.

[6] Portions of this chapter are published as:

Tse, E., Greenberg, S., Shen, C., Barnwell, J., Shipman, S. and Leigh, D. (2007) Multimodal Split View Tabletop Interaction Over Existing Applications. Proceedings of IEEE Tabletop 2007, 129-136.

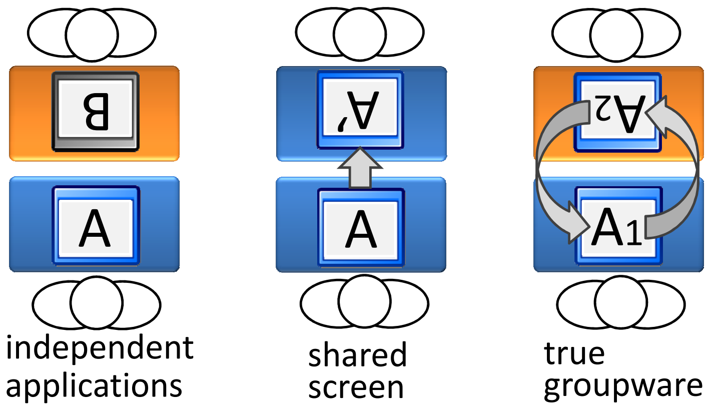


Figure 7.1. Software configurations for two people working face to face on a split view tabletop

7.1 Introduction

In everyday physical collaboration over a shared visual surface, people fluidly transition between working closely together (tightly coupled) and working in parallel (loosely coupled). The situation is somewhat different in the digital domain.

When people are geographically distributed, they routinely work together while viewing independent applications (Figure 7.1, left; A and B are different applications). Because they cannot see each other's screens and bodies, they use other channels (voice, instant messaging) to explicitly state what is visible on the screen. To improve this unwieldy situation, collaboration-transparent shared screen systems duplicated the output of one person's application so that it was also visible on distant screens (Figure 7.1, middle; A' is a duplicated view of A). Joint action was allowed by a wrapper that gathered and serialized people's input through a turn-taking mechanism, and then passed it onto the application [Greenberg, 1990]. This is essentially a what-you-see-is-what-I-see (WYSIWIS) view, where each person sees exactly the same visuals and fine-grained changes on their display [Stefik *et al.*, 1987]. Because of the strong linkage between views, shared screen views work well for tightly coupled work. Alternately, collaboration-aware groupware systems understood that multiple people were working in the space (Figure 7.1, right; A1 and A2 are instances of the same groupware application that are linked with one another). Collaboration-aware groupware systems facilitate loosely-coupled work by allowing simultaneous input, and by relaxing WYSIWIS to allow people to navigate and work independently on different parts of a large digital workspace [Stefik *et al.*, 1987].

Generally, the transition between loosely and tightly coupled work in distributed applications is enabled by the mechanics of collaboration [Gutwin and Greenberg, 2004]. People's awareness of each other's speech, gestures, and gaze actions produce consequential communication around the work surface that facilitates how they engage, interact, coordinate, and transition between loosely-coupled and tightly-coupled work. Common groupware awareness methods supporting these mechanics include telepointers for gesturing [Greenberg *et al.*, 1996], and radar overviews to give people a sense of what others are doing if they are working on different parts of scene [Gutwin and Greenberg, 2004].

Within the context of co-located groupware, things are somewhat analogous. The independent applications configuration of Figure 7.1 (left) happens when people seated next to each other are working on separate computers; their talk can draw attention to each other's display. As they turn to look at one of the screens, they have just transitioned to a simple shared screen system. The limitation is that there is only one input device, so only one person actually interacts with the system, or they manually share that device through turn-taking. To mitigate this, early single display groupware (SDG) systems provided multiple input devices so people could work in parallel [Stewart *et al.*, 1999]. There are two primary approaches to SDG:

1. ***Collaboration-Transparent SDG*** exploits how most operating systems merge the input from multiple mice into the single input stream seen by the standard single user application. The result is akin to screen-sharing (Figure 7.1, middle). While each person has a mouse, they still have to negotiate whose turn it is. That is, collaboration-transparent SDG favours very tightly coupled work through strict WYSIWIS turn-taking.
2. ***Collaboration-Aware SDG*** uses custom-built groupware applications that recognize and take advantage of multiple mice. Typically, all people have their own cursor and can work in parallel [Stewart *et al.*, 1999] akin to collaboration-aware groupware (Figure 7.1, right). However, the constraints of the small display usually mean that people are limited to a strict-WYSIWIS view, which in turn favours tightly coupled

work. More recently, the development of high-resolution digital walls and tables provide a large enough surface so that people can work somewhat more loosely-coupled when using collaboration-aware SDG: while they all still share the same view, they can work on their corner or side of the surface.

Obviously, collaboration-aware SDG is more flexible than collaboration-transparent SDG in supporting the spectrum of loosely- to tightly-coupled work. The problem is that very few existing real world applications are built as collaboration-aware SDG, thus limiting what people can do. On the other hand, collaboration-transparent SDG can immediately leverage existing applications, but is really amenable only to tightly-coupled work. The question is: can one exploit collaboration-transparent SDG in a way that allows people to work in a loosely-coupled fashion, while still giving them the ability to move towards tightly-coupled interaction by providing strong awareness cues of what the other is doing?

My answer is multimodal split view interaction, a split-screen tabletop configuration that supports both loosely and tightly-coupled joint work over conventional applications projected on a digital table, where the table also promotes awareness through multimodal interaction. The remainder of this chapter introduces this concept. I begin with split view interaction, and define three software configurations that constrain how it can be achieved. I continue by adding multimodal speech and gesture interaction capabilities to these split views, which enhance awareness during loosely-coupled work.

7.2 The Split View Tabletop

The first half of this system is the **split view tabletop (SVT)**. It is defined as a digital tabletop, where its surface is split into two adjacent projected views, and a person is seated in front of each view. SVT size expectations are that each person can easily see and reach into any part of their view. Seeing and reaching into the other view is slightly more difficult, but can be done by looking up, or by standing and leaning over the table. The basic idea is that people can work in a loosely coupled manner over their own individual views, and can also work in a tightly coupled manner either by shifting their attention to the other view or by linking their views together through software. Several key factors influence SVT: the actual software being projected, seating, size, and input devices used.

7.2.1 Projected Software

One of my goals is to work with existing 'conventional' applications rather than re-write applications from the ground up. If successful, SVT can be repurposed for myriads of available applications in co-located work. In this section, I describe three configurations that leverage existing software over SVT. Each is analogous to the distributed configurations described previously, so I reuse Figure 7.1 to illustrate each software view configuration in SVT. I show how each configuration offers different application sharing abilities, and how each supports different levels of collaborative work.

Individual applications allow people to work on their own separate application, each displayed on one side of the split view surface (Figure 7.1, left). For example, one person can use a web browser while the other person uses a digital map. The advantage is that people can work over separate applications in parallel yet still have some peripheral awareness of the actions of others simply by glancing up. The disadvantage is that both applications are not aware of each other, so that people have to resort to shifting attention and reaching over to one of the views if they wish to work in a tightly-coupled manner.

Screen sharing takes the screen displaying an unaltered single user application and projects it onto both views (Figure 7.1, middle) [Greenberg, 1990]. This can be implemented trivially using variants of the Virtual Network Computing (VNC) protocol [Richardson *et al.*, 1998]. As in the distributed setting, this forces a strict WYSIWIS view and sequential interaction through a turn-taking policy, making it also somewhat equivalent to *collaboration-transparent SDG.* Within SVT, screen sharing means that people can work very closely together over a single application, even though only a single person can interact at a time. Unlike individual applications, awareness cues arising from gaze and gestures are available.

Collaboration-aware groupware uses applications designed for distance-separated people, where it instead displays an application instance in each view on the digital table (Figure 7.1, right). This means I can exploit real-time distributed groupware within the co-located setting, which is akin to collaboration-aware SDG. This includes many PC games, as these are actually groupware designed to run over the Internet. This configuration naturally affords relaxed WYSIWIS, where people can work on their own part of the system without affecting others.

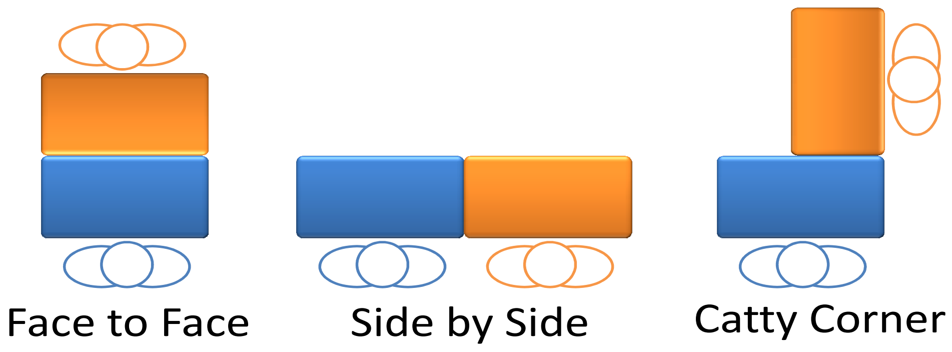


Figure 7.2. Seating arrangements for two people.

7.2.2 Seating Arrangements

Seating arrangements give different affordances to the SVT setting, where they can profoundly influence collaborative practice. Consider the following three common seating configurations.

Face to face seating provides people with easy visibility of each other's gestures and gaze actions on the work surface, as well as easy viewing of one another during conversation [Somer, 1969, Rogers and Lindley, 2004] (Figure 7.2, left). This is done simply by glancing up. This arrangement is commonly preferred for co-acting and conversation [Hall, 1966]. The cost is orientation, where studies have shown that displays that are text-heavy are significantly more difficult to read when upside down [Wigdor et al, 2005]. As well, a person can directly interact in the other view only by standing up and leaning over the table, or by actually moving to the other side of the table.

Side by side seating also affords visibility of each other and their work by side glances (Figure 7.2 middle). One can also reach into the other's view simply by sliding over. Orientation is not an issue as text is usually upright to both viewers. This arrangement is commonly preferred for cooperative tasks [Somer, 1969] as it easier to read text on a collaborator's display [Wigdor *et al.*, 2005]. The disadvantage is that side glances are more effortful than glancing up, and thus happen less frequently. As well, simultaneously viewing another person and the workspace at the same time is somewhat harder as the viewing angles are different. Finally, pairs are in very close proximity, and this could be discomforting if they do not know each other well [Hall, 1966].

Catty-cornered seating is a compromise (Figure 7.2 right). As with face to face, a glance up provides easy viewing of the other's work and their body. Its viewing angle offers slightly better text readability of their partner's screen [Wigdor *et al.*, 2005]. However, eye contact is harder to maintain when people are working over the surface. This seating arrangement is commonly preferred for tasks involving extended conversations as it supports the viewing of other's gestures while not requiring continual eye contact [Somer, 1969].

7.2.3 Size, Working Area, Reach, and Gaze

The physical size of each person's view can have a considerable effect on interaction [Hall, 1966, Somer, 1969].

Small views mean that people can easily reach into both spaces and that they can engage in each other's activity at a glance. The trade-off is limited working area, which is important for parallel work [Rogers *et al.*, 2004].

Arms length views are sized so that a person can just reach all parts of their own view while seated, yet still reach into parts of the other person's view if they stand up or lean over. This is a size where collaborators can still engage in each other's activity at a glance [Somer, 1969].

Large views afford even more space to work in parallel. The cost is that people cannot easily reach all parts of their view while seated, let alone the other person's space. Awareness is also harder to maintain as the other person's display is further away and details become difficult to see [Somer, 1969].

7.2.4 Input Devices for Pointing / Selection

Input devices for pointing and selecting influence not only on a person's individual work, but also how others can maintain awareness of the gestures of collaborators. Rather than catalogue the myriad of tabletop pointing devices and techniques that are under active development, I consider them as broad categories.

Decoupled devices such as a mouse or trackball are physically independent from the display surface. While efficient for individual work, other people may have a hard time noticing small device movements and button presses. A person will likely find it difficult to interpret what another person's activities mean unless he is looking directly at the telepointer [Greenberg *et al.*, 1996] or the artefacts being affected.

Distant freehand devices are directed at the surface but do not directly touch the spot that it is manipulating. Examples are systems that use ray casting or laser pointers on a large display. While the actions of others tend to be more visible, it may still be difficult to interpret one's activities.

Direct touch devices correspond directly with the part of the surface being manipulated. Examples include touch tables such as Smart Boards and DiamondTouch [Dietz and Leigh, 2001]. Within a split view setting, the direct engagement of these absolute devices maximizes one person's awareness of the other's activities and their meaning.

7.3 Multimodal Interaction for Awareness

The second half of the system is its multimodal speech and gesture interaction capabilities, provided both to facilitate a person's fluid interaction with applications on the table, and to promote awareness between working partners.

Existing applications displayed by the SVT system are almost always designed for a mouse, which is a decoupled device. That is, if people are working in a loosely-coupled manner, these small device movements will be hard for others to notice. For example, observational studies [Rogers *et al.*, 2004] of people working over a wall and table display using single user applications revealed that people maintain awareness [Gutwin and Greenberg, 2004] by "physically moving back to the table to be in close proximity" to other collaborators and using "outlouds to get the attention of others" and attract the attention of people on distant displays by "shouting out and giving directives to him/her as to what to do next."

For this reason I advocate multimodal speech and gesture commands that serve as both commands to the computer and as awareness for other collaborators (Chapter 3). Gestures create consequential communication of each other's bodies and activities, while speech serves as verbal communication to others. While single user applications do not understand gestures or speech, one can create gesture and speech wrappers (macros) that activate keyboard and mouse sequences that in turn invoke application functionality (Chapter 3). No changes of the underlying application code are required.

In Chapter 3, I used multimodal speech and gesture commands to enhance how people interacted over a digital table using single user applications. As described in Chapter 4, studies revealed that people exploited these commands for communicative purposes such as answering questions, validating understanding and agreement, and affirming statements made in prior conversations. People also used their improved awareness of each others actions to coordinate near-simultaneous activity by gracefully interleaving speech and gestures commands across people in the construction of commands.

7.4 Case Studies

I developed hardware and software to illustrate the multimodal SVT concept. Descriptions of the final systems are provided here, with implementation details deferred to §7.5.

The physical arrangement of the multimodal SVT implementation is illustrated in Figure 7.3, Figure 7.4, and Figure 7.5. I decided upon the face to face seating arrangement (§7.2.2) as I was most interested in situations where people moved from loosely-coupled to tightly-coupled work involving co-acting and conversation. I chose an arms-length physical size (§7.2.3) to balance reach and availability. For input devices (§7.2.4), I refactored a multimodal speech and gesture recognition system: GSI DEMO (Chapter 5). Gestures were recognized through a DiamondTouch multi-user touch surface [Dietz and Leigh, 2001] as a direct touch device (§7.2.4), while speech was recognized through headset microphones (§7.3).

The case studies below show sample implementations of all three software configurations (§7.2.1).

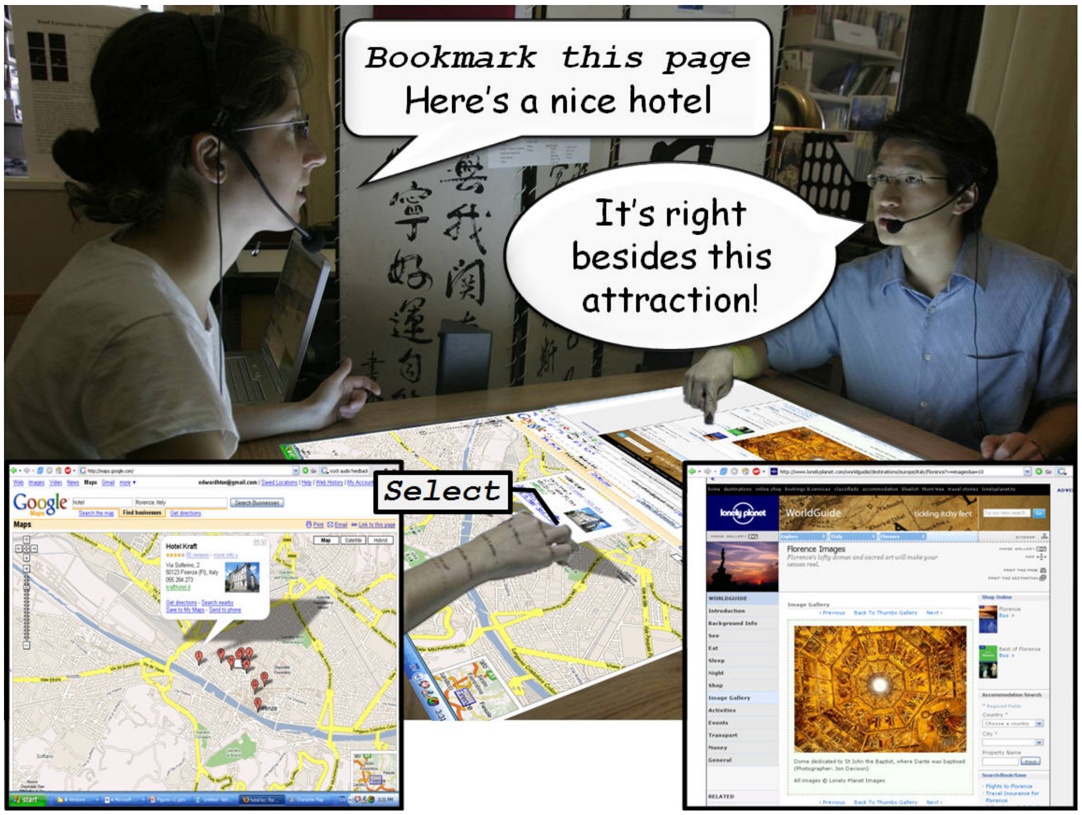


Figure 7.3. Split View over Independent Applications

7.4.1 Independent Applications

The independent applications configuration is appropriate when multiple people need to work independently over separate applications while still being aware of the actions of collaborators. While Section 5.5.2 explored how Mozilla Firefox could be mapped to speech and gesture actions for multiple people over a digital table, this mapping is quite limited in practice. Analogous to using a desktop computer, only one web page can be viewed at a time. Collaborators must take turns searching for complementary material. In contrast, Figure 7.3 shows two people planning a trip together by simultaneously browsing the web. Each runs a completely independent instance of the Mozilla Firefox web browser in their view. On the left (and in fuller view in the left inset), the woman is searching for Hotels using Google Maps. On the right the man is finding nearby attractions using an online Lonely Planet Guide.

Gestures across the seam are allowed. If the man drags a web link from his browser across to hers, that page is automatically loaded in her browser.

In Figure 7.3, the woman uses a voice command to bookmark the current page and a one-finger gesture to select the detailed information box of a hotel. She is simultaneously conversing, where she tells the man that the hotel could be a good one. The man responds by glancing over at her selection, and then referring to an attraction in his view that is close by that hotel.

As shown by this example, this configuration is good for loosely-coupled work during a joint task, where people can occasionally bring attention to some of their activities, e.g., by having the other person view it. While joint viewing is easy, joint interaction is difficult as one person would have to reach over the table to access the other view.

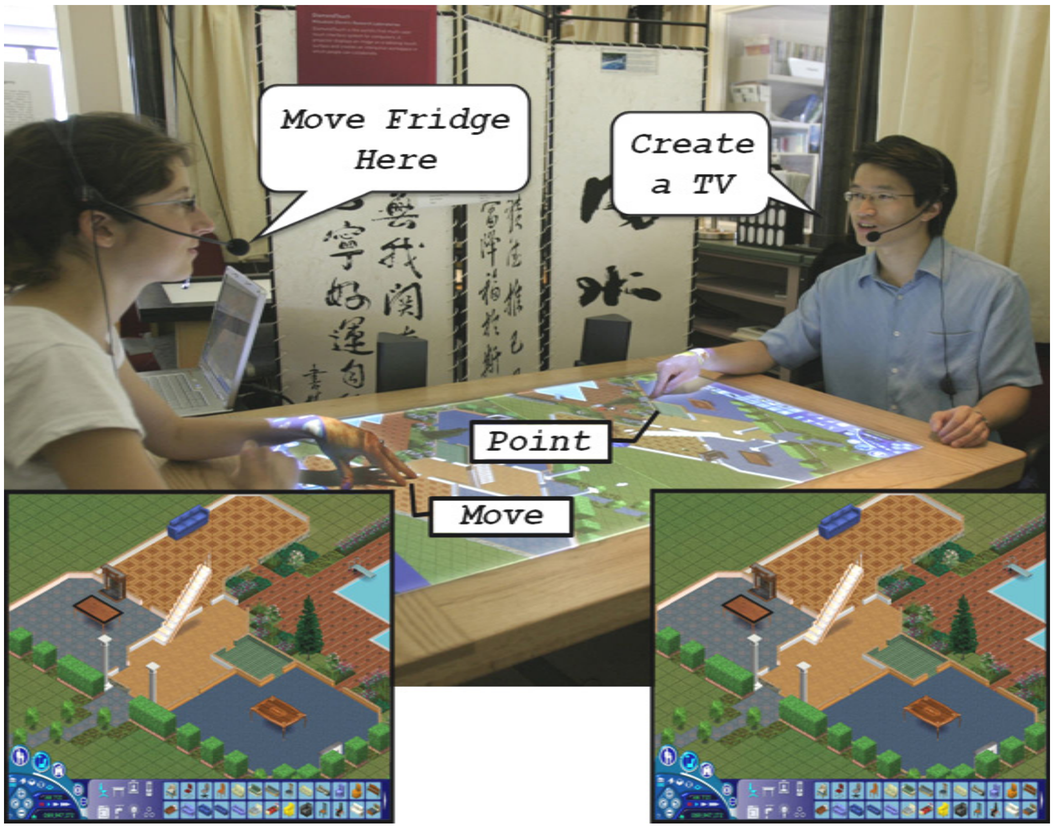


Figure 7.4. Screen Sharing using VNC and The Sims

7.4.2 Screen Sharing

The screen sharing configuration is appropriate when people need to work tightly coupled over the same view. While Chapters 3, 4, and 5 described how multiple people could use speech and gestures to interact with The Sims by Maxis, it was difficult to add furniture to the same room of the virtual home at the same time because people's hands and arms would often get in the way. Collaborators would often choose to work in separate rooms to maintain a distance from others over the digital table [Somer, 1969]. To help remove this constraint, Figure 7.4 shows two people interacting with a shared view of The Sims, where the views of both people are identical thus people can work over the same room at the same time without their hands and arms interfering with the other person. As well, the split view means that each person can reach the entire space.

I adapted The Sims wrapper described in Chapter 3 for the shared screen split view tabletop where one collaborator works on the actual application while the other works on a shared screen copy of that application. Similar to the wrapper described in Chapter 3, speech commands included "create <*object*>", "<*1st/2nd*> floor" while gesture commands included one finger placement, five finger movement and one fist object stamping. There were also multimodal commands that required both speech and gesture, such as "create <*object*> [one finger point]", which creates an object at the location being pointed to. As described in Chapter 4, this interaction is powerful; statements like "create a table [one finger point]" not only command the system, but also provide awareness to other collaborators about their actions. This is necessary for people to interleave their actions through turn-taking.

Figure 7.4 illustrates how this works in practice. The man is indicating through speech that he wants to create a TV while using a gesture to point at the spot while the woman is 'simultaneously' moving a fridge. The woman is working over the original application while the man is working over a shared screen view of the same application. Both commands are serialized and sent to the actual application as two independent commands.

A concern with screen sharing is that simultaneous interaction cannot be guaranteed. However, awareness minimizes conflict: when people know what the other is doing, they mediate their actions accordingly. Still, global actions can cause interference, e.g., if one person pans the map as another person is creating an object on a particular spot.

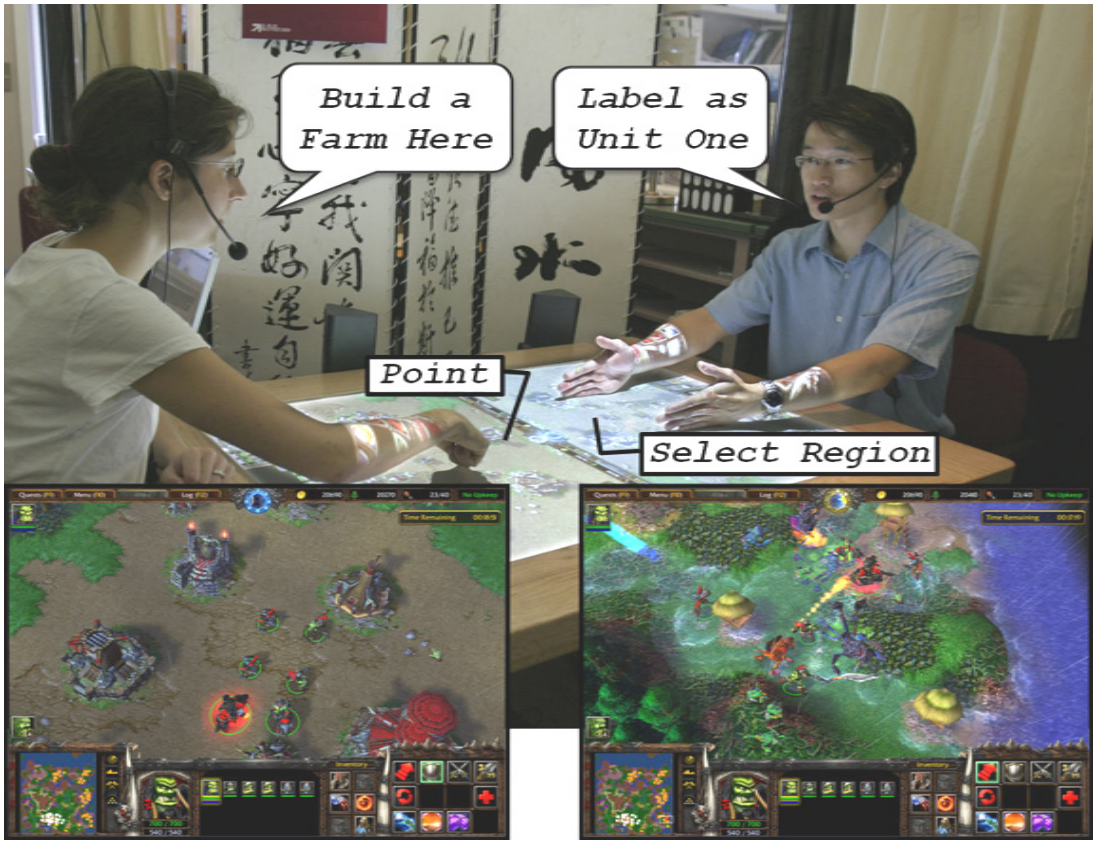


Figure 7.5. Collaboration-Aware Groupware using Warcraft III.

7.4.3 Collaboration-Aware Groupware

Collaboration-aware groupware is appropriate when people need to work both in a loosely- and tightly-coupled manner, and when their combined activities reflect changes in the common workspace. Chapter 3 described how multiple people could use speech and gesture commands over a single instance of Warcraft III, yet they were limited as people had to work over the same view of the terrain at all times. Compare this to Figure 7.5, which shows two people interacting with Warcraft III, a groupware game originally designed for distributed Internet players. As seen in Figure 7.5, the views of both people into the game's map surface can differ, yet both views share the same common game environment so actions by either person occurs in both places.

Akin to the mapping described in Chapter 3, I mapped speech and gestures to keyboard / mouse actions. Gestures included one finger unit selection, one hand panning and two hand side bimanual selection. Speech commands, some combined with gestures, include "[selection] label as unit #", "<*move/attack*> here [point]", "build <*building*>", and "stop". Players can also move troops across the seam by saying "move here [point]" and touching in their partner's view.

Figure 7.5 illustrates how this works as each person is pursuing duties on different parts of the scene. The woman is directing worker construction in one area through a speech/gesture action ('build a farm here [point]), while the man is directing troop actions in a region in a different area ('label as unit one [select region]). An overview map on the bottom left of each view shows the entire map surface, and immediately reflects each other's actions.

Leveraging distributed groupware to the split view setting is obviously powerful for it allows simultaneous action in a relaxed-WYSIWIS setting. Of course, people can also align their views to achieve WYSIWIS, although this has to be readjusted during panning (as scrolling is not synchronized). Awareness support within the distributed groupware system, such as radar views and feedthrough of other's actions in the scene [Gutwin and Greenberg, 2004], is enhanced by seeing the speech and gestural acts of others. People can leverage gestures over the seam, e.g., the man selects units in his view and moves them to the woman's view by saying "move here" and pointing to the woman's view.

7.4.4 Moving Between Configurations

SVT can be configured so that people can switch to a more conventional tabletop mode that presents a single view onto the surface. Ideally, they should be able to switch between variations of the configurations above. This part of the software remains to be implemented, but is straightforward. For example, the travel planners in Figure 7.3 could move into tightly-coupled work over Google Maps by switching from the independent application mode to a shared screen mode. If they want to work even more closely together, they can switch it again so only a single view of Google Maps is shown across the entire table. Similarly, the Warcraft players in Figure 7.5 can move from a collaboration-aware groupware view into a shared screen or the single view configuration if they wish to move into tightly-coupled strict-WYSIWIS interaction. Alternately, one player can bring up a different view as an independent application, e.g., to look up a cheat sheet for Warcraft III on the Internet.

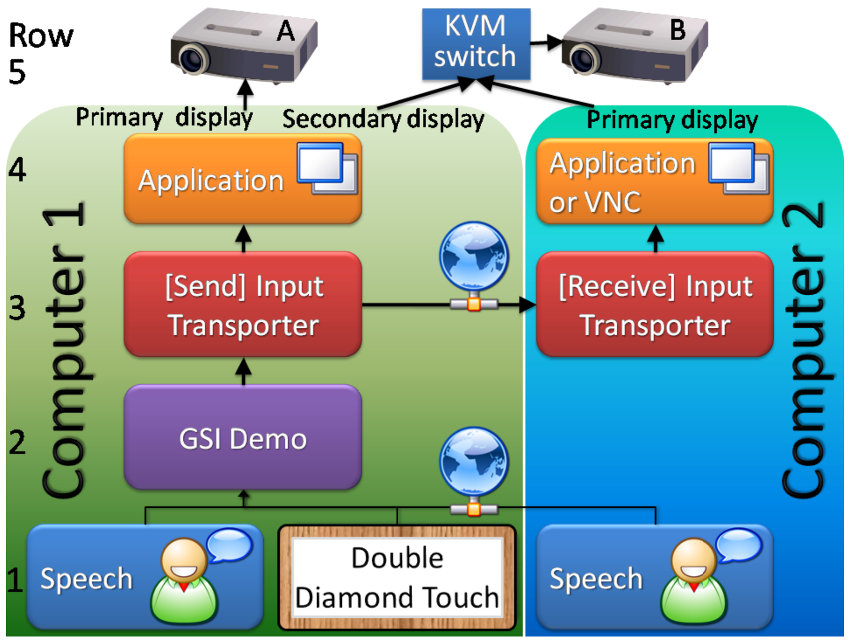


Figure 7.6. The SVT Infrastructure

7.5 Implementation

Multimodal SVT interaction over existing single user applications is a new concept, yet I was able to build this environment by repurposing various hardware / software systems at my disposal (Figure 7.6).

7.5.1 Hardware[7]

Hardware comprises 2 touch-sensitive tables, 2 projectors, 2 computers, and speech input.

The split view digital table. The digital table seen in Figure 7.3, Figure 7.4, Figure 7.5, and Figure 7.7 is called the Double DiamondTouch (Figure 7.6, row 1, Figure 7.7) we designed as a face to face (§7.2.2) and arms-length (§7.2.3) surface. The table's total size is 148 x 116 cm, comprising an active area of 98 x 65 cm (117cm along the diagonal axis) and a 25 cm wide solid oak bezel. Unlike the DiamondTouch surfaces (DT 81 and DT 107)

[7] The Double DiamondTouch was done in collaboration with John Barnwell, Sam Shipman and Darren Leigh from Mitsubishi Electric Research Laboratories.

described in earlier chapters (Figure 7.7), the Double DiamondTouch has a different aspect ratio (3:2 for Double DiamondTouch versus 4:3 for DT 81 and DT107) since the active area is made of two smaller DT81 DiamondTouch surfaces [Dietz and Leigh, 2001] laser cut to produce almost no seam (or break) between the two surfaces. The oak bezel and its beveled edges provide a comfortable resting area for people's arms that would not unexpectedly perform an action on the display surface. The beveled corners were designed to prevent arm strain over extended use.

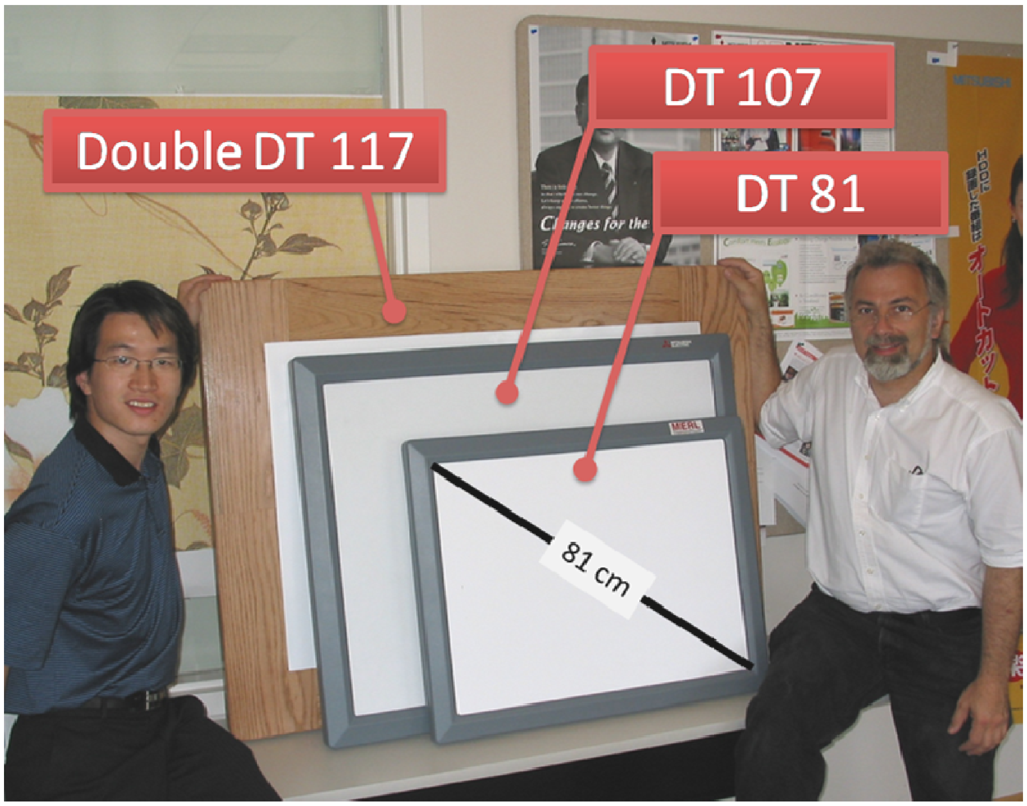


Figure 7.7. Size comparison of the Double Diamond Touch

Projecting multiple views. We used two projectors (Figure 7.6, row 5) and two computers (Figure 7.6, left and right half) to top-project two views onto the table, one in each half. Two 1024x768 LCD projectors give a total resolution of 1536x1024.

Switching. As described in §7.4.4, one can switch between a single large shared display from one computer to two separate views from two computers. To do this, we used a KVM (Keyboard, Video, Monitor) switch (Figure 7.6, row 5). Projector A is connected to the primary display for Computer 1 while Projector B is connected to the KVM switch. The KVM switch is connected to both Computer 1's secondary display and Computer 2's primary display. Thus toggling the switch either displays Computer 1's contiguous view across the entire table, or Computer 1 and 2's view in the corresponding split views. The orientation of the projected image is adjusted so that it is the right way up for the seated viewer. This manual switching process is automated, where we use Phidgets [Greenberg and Fitchett, 2001] to programmatically control the KVM switch. A person uses simple voice commands "split view mode" or "shared view mode" to transition between the two.

Detecting touch input from multiple people. The DiamondTouch provides the necessary multi-user simultaneous touch input [Dietz and Leigh, 2001], as well as a reliable way to uniquely identify which person belongs to each touch. This is necessary to disambiguate who is doing what in the SVT environment. Both DiamondTouch surfaces are connected to a single DiamondTouch hardware controller board that has modified firmware to handle the larger board size. To the end programmer, the Double DiamondTouch appears as a single large DiamondTouch surface. All its input is sent to a single computer (Computer 1 in Figure 7.6) where a driver on that computer receives that input so that it can be used in application development.

From a firmware perspective, both Diamond Touch surfaces are connected to a single Diamond Touch hardware controller board that has modified firmware to handle the larger board size. The Diamond Touch surface works by rapidly scanning through a large number of horizontal and vertical transmitters located on the touch surface. By doubling the number of antennas we effectively halve the speed of the input device. The Double Diamond Touch runs at an input speed of 25Hz. All input is sent to a single computer (Computer 1 in Figure 7.6) through a USB connection. Special software drivers are required to access input from the Double Diamond Touch so that it can be used in application development. To the end programmer, the Double Diamond Touch appears as a single large Diamond Touch surface.

Detecting speech input from multiple people. We used the technique described in Chapter 5 to gather simultaneous speech input. Two Labtec LVA 7330 noise cancelling microphones are each connected to off the shelf speech recognition software. Recognized speech is then sent to a single computer for further processing (Computer 1 in Figure 7.6).

7.5.2 Software

Software is built atop GSI DEMO (Chapter 5), a system originally developed for a single display multimodal tabletop surface (Figure 7.6, row 2. With GSI DEMO, people program by demonstration to map speech and gesture actions onto keyboard and mouse events for multiple people. After training, GSI DEMO listens for people's speech and gesture commands, and invokes the appropriate mouse/keyboard counterpart.

Processing input. I consolidate the speech and gesture input of multiple people to a single computer so that one can process it more easily. This makes some tasks, such as turn taking management, much easier. Figure 7.6 shows the SVT software infrastructure behind this. Input from multiple microphones and the Double DiamondTouch (Row 1) are eventually received by the single computer using GSI DEMO (Row 2) described in Chapter 5. GSI DEMO then plays back the appropriate keyboard/mouse actions as if the user had entered them (Row 4).

As an aside, the original GSI DEMO was developed for adding multimodal input atop a single display. In multimodal SVT, GSI DEMO now mediates input and output from multiple computers. To do this, it uses the distributed shared dictionary data structure provided by the GroupLab Collabrary [Boyle and Greenberg, 2005] to send and received speech and gesture input to the appropriate computer (Figure 7.3, Row 3). Events sent include: speech volume /recognition/hypothesis and gesture down/move/up events.

Mapping input to screen coordinates. The raw input coordinates of a gesture cannot be used directly, as input is provided in table coordinates. This differs from the screen coordinates for either computer. For example, the top left of Computer 1's display does not correspond to the top left of the table. To solve this, I created input mapping rules that convert gesture coordinates into screen coordinates. These rules consider the orientation, resolution and size of each display. In particular, the input transporter (Figure 7.6, Row 3) uses a configuration file to set the seating arrangement, display and software configuration. Using this information, all transported input is converted to screen coordinates for each computer using a simple linear transformation.

Mapping speech / gesture to keyboard / mouse events. To implement screen sharing, the input transporter creates a VNC-like system [Richardson, 1998], which sends screen snapshots captured by the GroupLab Collabrary [Boyle and Greenberg, 2005] every 100 milliseconds to all client computers (Figure 7.6, Row 4). All input is serialized by the input transporter and is passed on as keyboard / mouse events to the computer running the single application. GSI DEMO also includes several turn-taking protocols that mitigates interference when people try to work simultaneously [Greenberg, 1990, Tse *et al.*, 2006a, Tse *et al.*, 2006c].

For both collaboration-aware groupware and independent applications configurations, the appropriate speech / gesture map is loaded onto each machine. Input for each computer is then managed as if it originates from each of the DiamondTouch surfaces that correspond to a particular computer, i.e., input from a particular view is passed onto the appropriate computer running the groupware instance or independent application.

In all cases, the single user applications shown in Figure 7.6, Row 4 receive simulated keyboard and mouse events and are completely oblivious to the use of speech, gestures and transported input.

Handling input across the seam. The Input Transporter API lets a programmer map custom actions onto drag or touch actions across boundaries. For example, if two desktop systems are running, the programmer can create a mapping that detects if a file is dragged across the boundary and then invoke a 'copy file' onto the desktop of the other computer.

7.6 Related Work

There is a long history of research in distributed groupware [Greenberg *et al.*, 1996, Gutwin and Greenberg, 2004], shared screen systems [Greenberg, 1990, Li and Lu, 2006], single display groupware [Stewart *et al.*, 1999], large digital tables and displays [Dietz and Leigh, 2001, Rogers and Lindley, 2004, Tandler *et al.*, 2001, Wigdor *et al.*, 2005] and multimodal interaction [Bolt, 1980]. However, several works stand out in regards to multimodal split view tabletops.

Within collaboration-transparent screen sharing context, turn-taking protocols in distributed shared screen applications [Greenberg, 1990] and more recently in the large display SDG setting [Tandler *et al.*, 2001, Ringel-Morris *et al.*, 2004] regulate and limit how one person can interfere with another's activity. Simultaneous interaction with existing applications can be simulated if commands are combined into unitary chunks and then interleaved with others. This has been done in various SDG systems using PDAs [Myers *et al.*, 1998] and multiple mice [Stewart *et al.*, 1999].

There are other systems related to split screen tables [Rogers *et al.*, 2004]. Within the gaming world, there are a plethora of multi user console and arcade games that split a single screen, where each person works in their own view. Unlike my generalized solution, these games are typically implemented as special-purpose collaboration-aware groupware. Within the office productivity world, many applications let its user split a document into two independently scrollable views. Almost all are limited to a single point of input, although Li and Lu [2006] produced an application that could leverage a split view for multi user simultaneous input. Sing, Gupta and Toyama (of Microsoft Research India, research.microsoft.com/research/tem/) split a single computer display vertically: each person independently works on their half using their own keyboard and mouse (equivalent to independent applications).

Finally, two separate personal devices can be connected into a single display. With Connectables [Tandler *et al.*, 2001], two people move their small tablet displays into close proximity: sensors notice this and combine them into a shared workspace [Tandler *et al.*, 2001]. Alternately, force sensors detect what sides of two tablets bump against each other, and use that information to combine and adjust the orientation of the view automatically [Hickley, 2003].

7.7 Conclusion

In this chapter I explored the design space of multimodal split view tabletop interaction over existing single user applications. I offered a generalized approach to leveraging three types of existing software over a Split View Tabletop: independent views, shared screens and collaboration-aware groupware. I also added multimodal input as a way to promote awareness between collaborators. Three proof-of-concept systems illustrated how this works in practice.

This chapter presented multimodal split view tabletop interaction as a design concept that can be appropriate in particular collaborative situations. I do not expect multimodal SVT to replace conventional single view digital tables, rather it offers alternate configuration that could be used as particular situations warrant it.

While I did not do a formal evaluation of this system, limited use by colleagues revealed that the multimodal command mapping had a large influence in parallel work. That is, speech commands would often follow sequentially as people preferred not to talk over each other. This is likely due to the saliency of speech commands as users of this system had commented on how their awareness of others' speech actions influenced their own behaviour. These informal observations indicate that designers should consider the tradeoffs between parallel work and awareness when creating multimodal co-located systems. Of course, further investigation is required to confirm these claims in practice. In the following chapter I explore design issues encountered in building a collaboration-aware multi user multimodal application from the ground up.

Chapter 8. Exploring Collaboration-Aware Multi-User Multimodal Interaction

***Objective Seven**. I will develop a collaboration-aware multimodal co-located system to explore issues for future designers.*

Previous chapters focused on allowing multiple people to interact with collaboration-transparent wrappers over single user applications. However, simulating a single stream of input limits the types of multi user multimodal interactions that a designer can work with. In this chapter, I explore design possibilities in a custom application designed to support simultaneous speech and gesture input over a digital table. This collaboration-aware multi-user, multimodal interaction lets co-located people work in parallel or cooperate in using an application. It also allows the system to leverage content created within an application through handwriting recognition and allows the input of multiple people to be combined into a joint multimodal command. [8]

In this Chapter I explore the design space of collaboration-aware multi user multimodal interaction through a case study, where I implemented an application that supports the KJ creativity method used by industrial designers. Four key design issues emerged that have a significant impact on how people would use a collaboration-aware multi-user multimodal system. First, **parallel work** is affected by the design of multimodal commands. Second, individual **mode switches** can be confusing to collaborators, especially if speech commands are used. Third, establishing **personal and group territories** can hinder particular tasks that require artefact neutrality. Finally, timing needs to be considered

[8] Portions of this chapter are published as:

Tse, E., Greenberg, S., Shen, C., Forlines, C., and Kodama, R. (2007) Exploring True Multi-User Multimodal Interaction over a Digital Table, Proceedings of ACM DIS Conference (Feb 25-27, Cape Town, South Africa).

when designing **joint multimodal commands**. I also describe a model, view, controller architecture for collaboration-aware multi-user multimodal interaction.

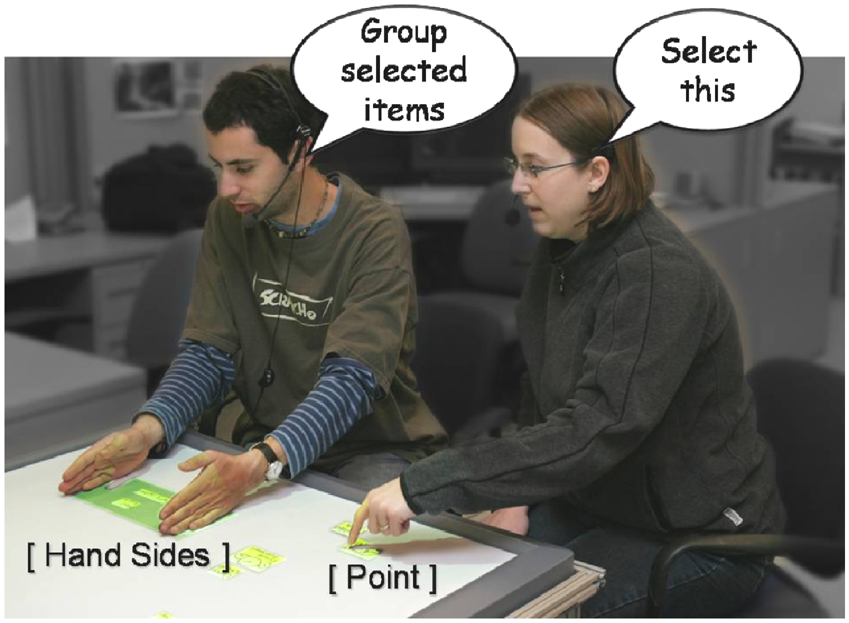


Figure 8.1. A two person grouping hand gesture

8.1 Introduction

In Chapter 3, I explored how existing off-the-shelf single user applications can be wrapped in a way that allows multiple people to interact with it over a digital table via speech and gesture commands. In Chapter 4, I described a study that revealed how these speech and gesture commands can serve as both commands to the computer and as communication to other collaborators. Yet these collaboration-transparent wrappers are limited by the one user per computer assumption of the underlying application and operating system. Multiple people cannot work in parallel because the computer can only accept a single stream of input.

In contrast, *collaboration-aware* multi-user multimodal interaction recognizes that multiple people are interacting with it, and allows co-located people to simultaneously gesture and speak commands to the groupware application (Figure 8.1). I explore the design space of such systems through a case study, where I implemented an application – the Designers' Environment – that supports the KJ creativity method as used by industrial designers.

To preview what is to come, the KJ creativity method has four basic steps: creating notes, grouping notes, labelling notes, and relating notes. The Designers' Environment supports these four basic steps by **idea sketching** onto digital notes, **grouping** using hand gestures, voice selections, and multimodal selection, **annotation** using handwriting, and **relating** by using sizing gestures or by linking notes with a multimodal command (Table 8.1). Four key design issues arose during system development.

1. **Parallel work.** The design of multimodal commands can greatly influence collaborators' propensity to engage in parallel work. For example, if the majority of commands were via the speech channel, people may be unwilling to talk over each other, which in turn would favour sequential work.
2. **Mode switching.** While a collaboration-aware multi-user multimodal system provides the potential for independent modes of action (e.g., one person is annotating while another is moving artefacts) confusion arises when people forget what mode they are in. This problem is exacerbated by publicly seen and heard multimodal commands that can give others the false impression that they are all in the same mode.
3. **Personal and group territories.** The design of multimodal commands can significantly influence the establishment of personal and group territories in the collaborative workspace [Kruger *et al.*, 2004, Scott *et al.*, 2004]. By paying attention to how speech and gesture commands are used, designers can develop systems to support or hinder the establishment of personal and group territories.
4. **Joint multimodal commands.** The multimodal interactions of collaborators can be combined into a single joint action (e.g., the joint actions of Figure 8.1, where one person is extending another person's selection). These commands need to carefully consider the time window for joint inputs to be recognized, or erroneous command additions and omissions may result.

The next section briefly introduces the KJ creativity method. I then describe the Designers' Environment system and detail the four design issues described above. I conclude by describing my implementation and examining related work.

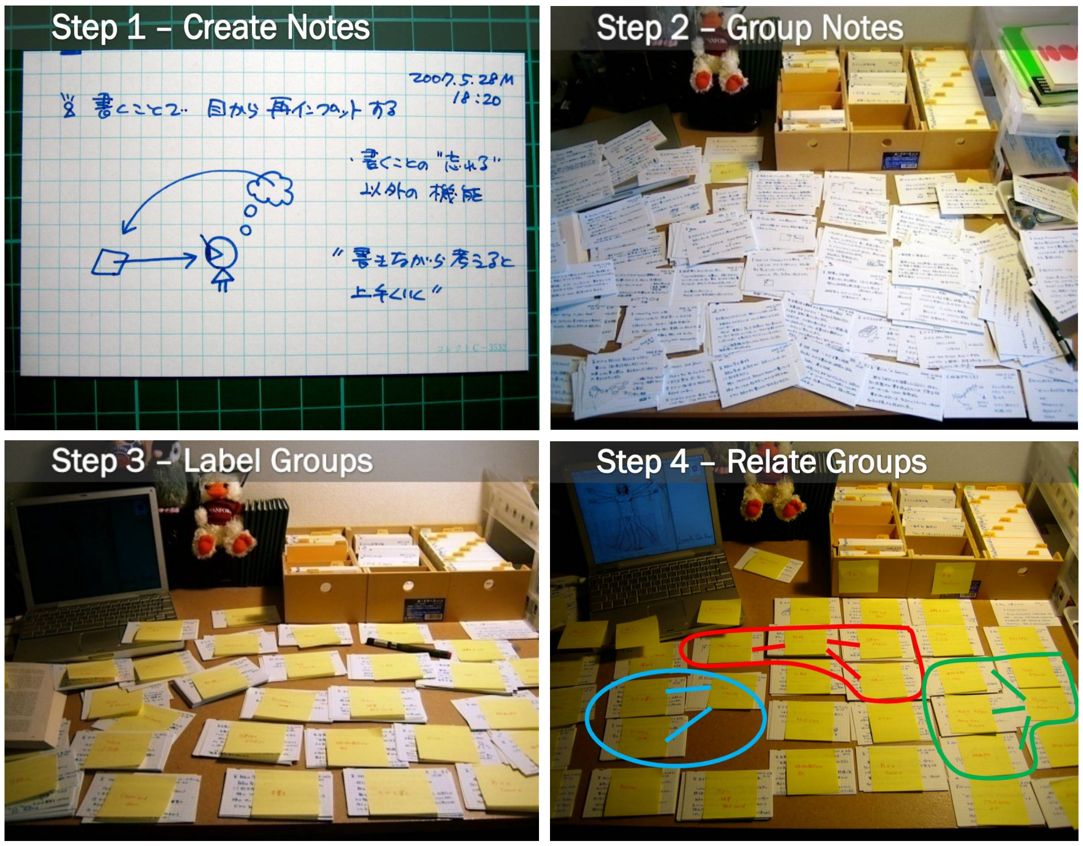


Figure 8.2. The four steps of the KJ creativity method, images from http://www.flickr.com/photos/hawkexpress

8.2 The KJ Creativity Method

I was approached by a group of industrial designers from a consumer electronics company to develop a system to improve their initial brainstorming activity via the KJ Method. Designers and marketers typically use the initial phases of the KJ Method to collaboratively brainstorm ideas and concepts for new products, to establish customer needs and to explore potential product features. The output of the KJ Method is a list of core needs and features that will be later used and refined by designers in their product sketches.

The paper version of the KJ Method is composed of four basic steps as illustrated in Figure 8.2. First, multiple people write customer needs, product ideas and comments onto 4x5" cards (Step 1). Each card can be brief with only a title, or it can provide additional details such as a description and illustration. Second, each card is randomly distributed by either shuffling the cards to each collaborator or by shuffling the cards around a table and having each person work on the cards that are closest to them. They then group together similar ideas into piles (Step 2). Third, piles are labelled according to the need/idea that they represent (Step 3). Finally, collaborators relate the notes by drawing links between groups and creating meta-groups representing common themes (Step 4).

Table 8.1. The Designers' Environment speech/gesture interface

Speech/Multimodal commands		**Gesture commands**	
[point to note] **Show recognition**	Reveals the text recognition result for the selected note	**One finger (on a note/group)**	Moves note or group
Recognize this *[note] /* **all notes**	Converts a note into text	**One finger (Empty space)**	Creates a lasso group
[point to note] **Find related images**	Opens a web browser with the note text as a search query	**One finger (annotation mode)**	Draws labels and links
[point to note] **Convert to note**	Converts a web items into an image	**Two fingers**	Zoom note/group
Annotation Mode	Allows one to draw links/create labels	**Five fingers**	Moves group or workspace
[select group] **delete this**	Deletes the selected group	**One hand**	Erases annotations
Select *<say note text> /* **this** *[note]*	Selects a note	**Two hand sides**	Create group between hands
Group selected items	Converts selected notes into a group	**link this** *[note]* **to this** *[note]*	Creates a link for two notes
[select group] **Arrange / Sort Alphabetically**	Tiles items within a group (sorted alphabetically)	**zoom this** *[note]* **/ all**	Zooms the camera to a note / all notes
Restore *[group]*	Returns items in a group to their original position	**Make this** *[group] <say a colour>*	Changes the group colour

8.3 The Designers' Environment

As a case study of a collaboration-aware multi-user multimodal system, I designed and implemented a groupware system for the KJ method that lets co-located people work together over a digital table and personal tablet PCs. This system is called The Designers' Environment, and we see two people using it in Figure 8.1. Multiple people create, group, label and relate digital notes using speech and gesture commands; these are summarized in Table 8.1. In this section, I describe the physical form factor and interactions for each step in this process, and how I leveraged the capabilities of collaboration-aware multi-user multimodal interaction.

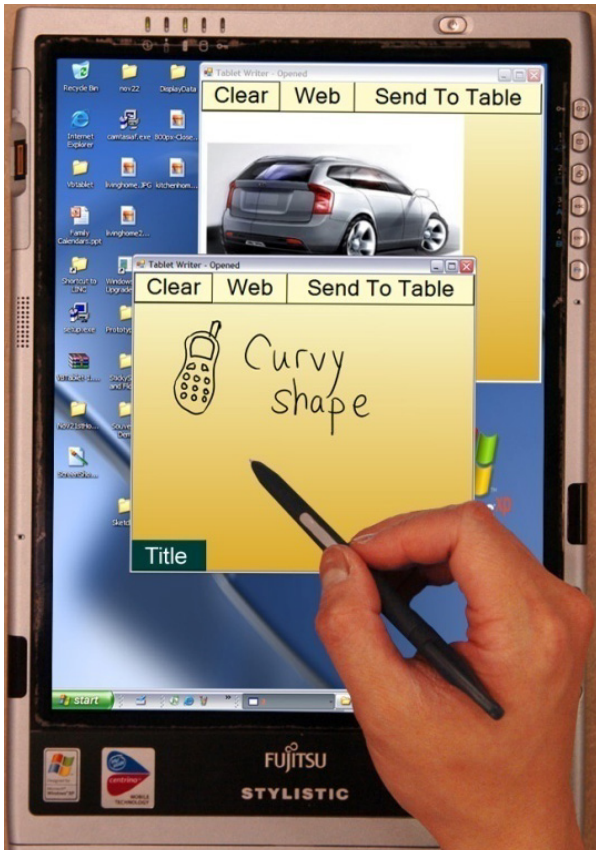


Figure 8.3. Tablet note writing

8.3.1 Creating Notes

Notes are the basic unit of the KJ Method. As shown in Figure 8.2 (Step 1), people create paper notes on 4x5" cards. Similarly, the Designer's Environment supports the creation of basic digital note. However, it also allows people to use a note's contents to search the web for related images and information, and to create new notes based on search results. Multiple people can do all these actions in parallel.

The basic note. Each participant independently creates digital 4x5" cards through a pen-based Tablet PC running a note writing application (Figure 8.3). They can quickly sketch and hand-write ideas, needs, descriptions and illustrations onto this card. When complete, they send the card to the digital table by tapping a 'Send to Table' button (Figure 8.3, top right). This automatically places the note in a random location on the digital table, thus mimicking the shuffling of cards of the KJ Method.

Searching and importing images and web pages. People can also import their digital images or snapshots of web pages into a note; while not part of the KJ method, this extra information could help people's discussions. Images and web page content capture experiences, emotions, and concepts that may be hard to express through words or illustration [Lucero and Martens, 2005].

People can use two methods to achieve this. First, if the person already has a saved image handy, he or she can drag a previously saved image into a 4x5" card, preview and optionally resize it, and then send it to the table as before (Figure 8.3, the note in the background). Second, a person can search for appropriate web content for related images or web pages and import those. One hand writes the search terms on a note, and then taps a 'Web' button (Figure 8.3, centre) to begin the search. Handwriting recognition translates the writing into machine-readable text, and feeds the result into Google Image Search. The web page of results is displayed. At this point, one can continue to navigate the web to a particular web page or image by navigating links. Clicking the 'Web' button a second time captures the image or web page on the 4x5" card, which can then be moved to the table. Once on the table, all images can be resized as needed.

Recognizing note contents and using it for searches. After a note is added to the table, its text is automatically processed by a handwriting recognizer and the result is stored as meta-data along with each note. People can reveal this data by several means. Pointing to a note and saying "show recognition" temporarily raises a popup containing the recognized text. Alternately, one can transform one or more handwritten note into text by saying "*[point to note]* recognize this", or saying "recognize all notes".

People can also use the note's meta-data of recognized text to search for information related to a note's contents. One searches for related images through the multimodal command "*[point to note]* find related images"; this triggers a web search on Google Images using the terms in the recognized text. That page is projected onto the digital table. As was done on the tablet, the person can then follow links until a desired web page or specific image is found, and then convert that into a new note by saying "convert to note".

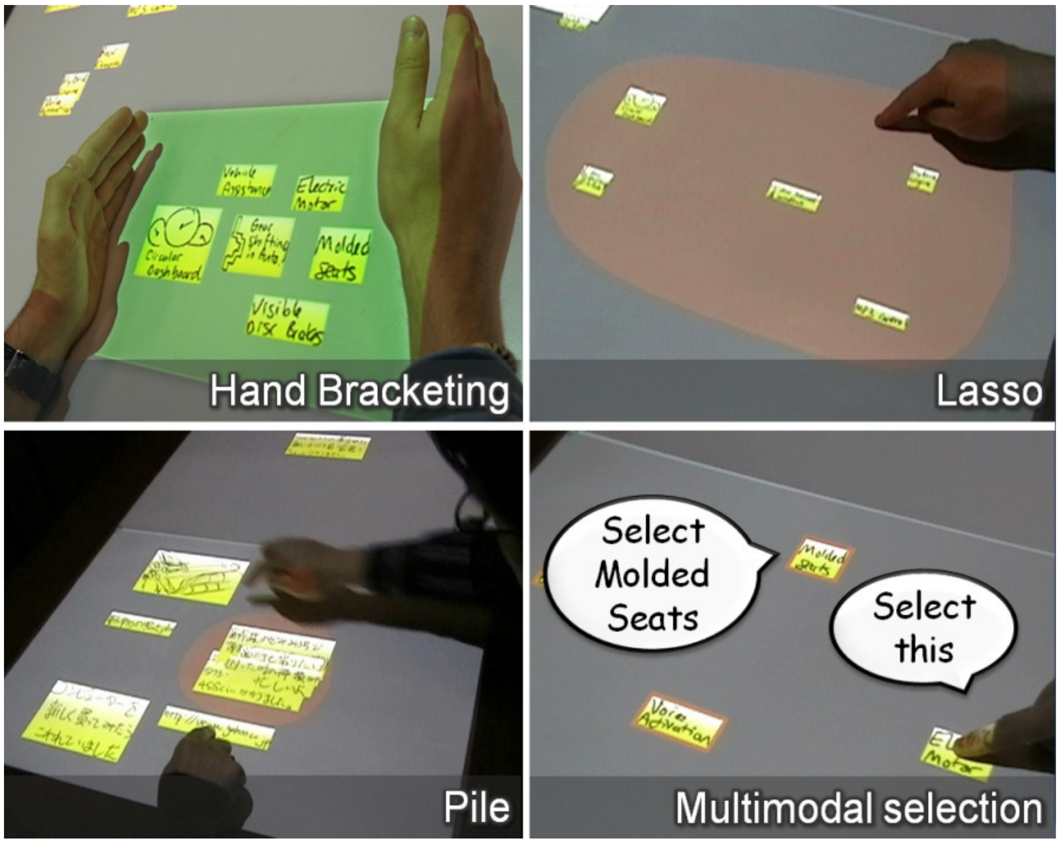


Figure 8.4. Grouping interactions

8.3.2 Grouping Notes

The second step of the KJ method is to group or pile related notes on the table (Figure 8.2, Step 2). Grouping is supported through several gesture and speech actions on the digital table, as illustrated in Figure 8.4 and described below. To encourage discussion and coordination between collaborators, all grouping is done on the digital table rather than through the Tablet PCs. All grouping actions can be done simultaneously by multiple people.

Groups are visually represented using a light highlight color (red, green, blue and tan). Each highlighted colour represents the individual that created the grouping. This makes it clear who created each group, which can help facilitate later discussion.

Hand bracketing and lasso grouping. People naturally explore item grouping by moving related notes next to each another. They do this on the table by using a single finger to move either single notes or previously established groups. Participants can then explicitly group each note by either using two hands to bracket an area (Figure 8.4, top left), or by using a single finger to draw a lasso around the desired notes (Figure 8.4, top right). In both cases, notes within the contained area are automatically included in the group selection. As an aside, notes can overlap so they look more like piles (Figure 8.4, bottom left). These groups can then be moved around the table by using five fingers (a grabbing gesture) or by using a single finger on an area within the group that does not contain a note.

Alternately, an empty group or pile can be created by lassoing or hand bracketing an area containing no notes. Notes can be later dragged into this area, which automatically includes them in that group. Empty groups can be deleted using the "[select group] delete this" multimodal command.

Searching notes by speech. Sometimes, people may want to find and select a note that is out of reach, covered by other notes, or lost in the clutter. To help in these cases, one can find notes using speech. Recall that a note's handwritten contents are recognized and stored as meta-data (§8.3.1). Under the covers, this meta-data is also automatically added to a speech recognizer. When a person says "select <*say note text*>", the note that best matches that text is highlighted with the users default colour (Figure 8.4, bottom right). Currently, the recognition system works best if the entire note contents are spoken verbally. This is hard to do for notes containing large descriptions, and it does not work over images and illustrations. A future system would benefit from a better speech search system.

Multimodal selection and grouping. Sometimes, people may want to select and group distant notes that are scattered around the table. They do this by first selecting one or more notes with speech (described earlier) or by doing a multimodal selection: "*[point to note]* select this", and then saying "Group selected items". All selected notes are then moved into a single neatly arranged group. People can collaboratively extend each other's selection, as illustrated in Figure 8.1.

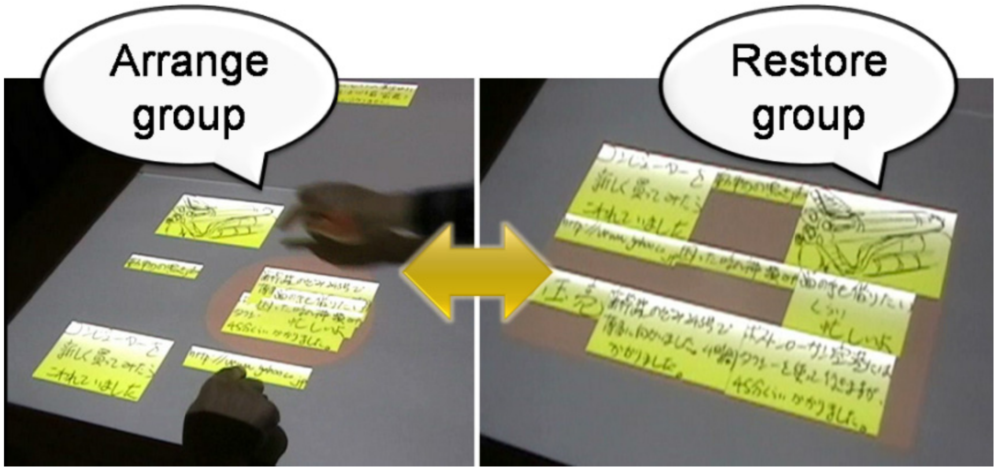


Figure 8.5. Expanding and collapsing groups

Rearranging groups. Next, people can rearrange notes in a groups through multimodal speech commands, either to reveal the hidden content of overlapping notes (as in Figure 8.5, left) or to bring distant notes within a group closer together (as in Figure 8.4, top right). When one says “Arrange group”, a smooth animation sequence re-arranges these notes as non-overlapping adjacent tiles as seen in Figure 8.5 (right). Arranged notes can be returned to their original location by using the “Restore group” speech command (Figure 8.5, right).

Sorting notes within a group. The “Sort alphabetically” speech command rearranges a group’s notes into alphabetical order. As before, the “Restore group” command returns the group to its original position. Non-textual illustrations appear at the top left corner of a sorted group.

Panning / zooming. As an aside, there may be insufficient space on the table to arrange all notes and groups. Consequently, I allow the entire workspace to be panned using a 5-finger gesture, and zoomed with a 2-finger gesture in an empty area (§8.5.3).

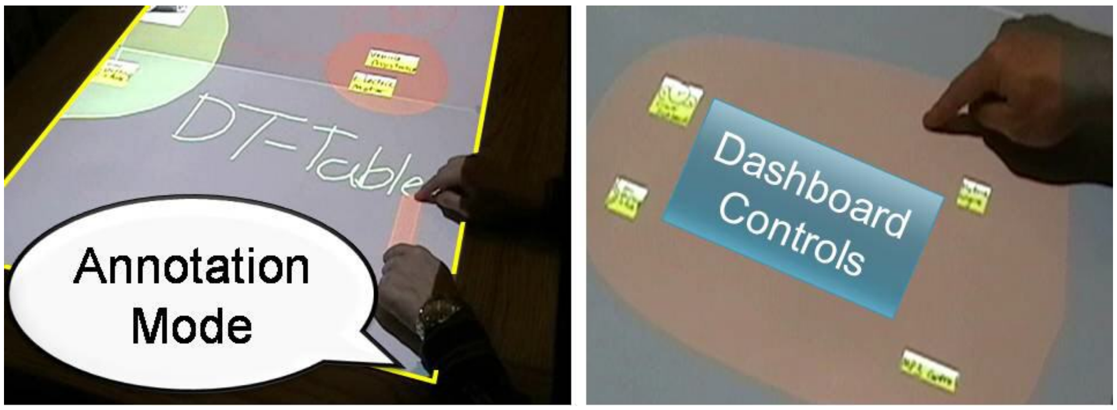


Figure 8.6. Group annotation methods: table annotation (left), adding a group title (right)

8.3.3 Labelling Groups

The third step in the KJ Method is to label groups and areas with a descriptive name (Figure 8.2, Step 3).

Labelling groups is supported in two ways. The first method is to write the label directly on the digital table surface using a finger (Figure 8.6, left), and the second is to create a special title note through the Tablet PC (Figure 8.6, right) and add that to the table.

Tabletop labelling. A person creates a textual label (and other marks) on the digital table by saying "Annotation Mode". A yellow border appears around the digital table indicating that a mode change has occurred for all users. From that point on any person can write and draw on the table with their finger, or erase through a whole hand erasing gesture. Saying "Annotation Mode" again exits that mode. Marks always appear atop other table artefacts, i.e., notes and groups. The restriction is that tabletop annotations cannot be moved. If a group is repositioned, any corresponding tabletop annotations must be erased and drawn again. While restrictive, tabletop labelling is at its best when regions of the table space can be split into theme areas.

Tablet labelling. Alternately, people can use the note writing application on the Tablet PC to create special 'title notes' that can become incorporated within a group. One switches to title notes by pressing a title note toggle switch (Figure 8.3, lower middle); the color of the note changes to dark aqua with white sketching, as with normal notes the person creates the label and presses the "Send to Table" button. Once on the table, a title note can be placed in a group and moved along with it. However, title notes have special features. They are always placed on top of other notes (Figure 8.6, right). When groups are re-arranged, either spatially or alphabetically, title notes are always placed as the first item in the top left corner.

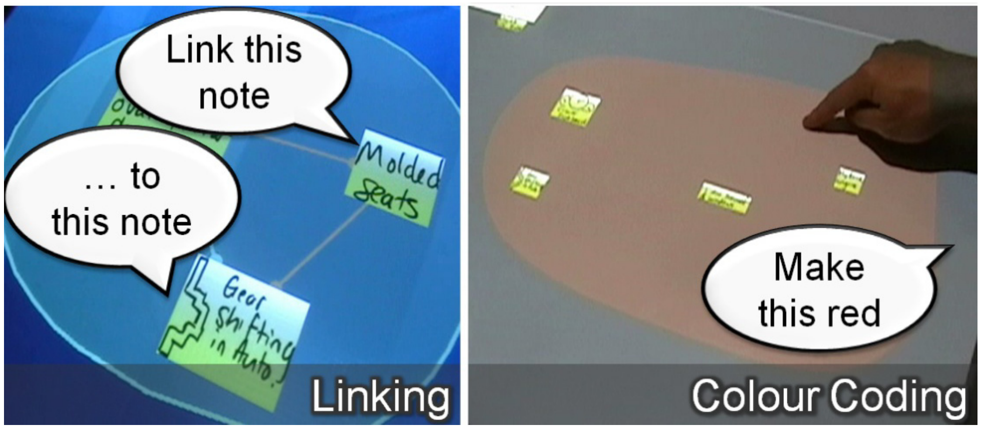


Figure 8.7. Two methods for relating notes and groups

8.3.4 Relating Groups

The final step of the KJ Method is to relate groups (Figure 8.2, Step 4). Typically, people draw links between related notes, and spatially create meta-groups that represent common themes. In the Designers' Environment, people can perform these actions through public speech and gesture commands over the digital table. First, related groups can be moved closer together. Second, meta-groups can be created and links between contained groups can be drawn. Third, groups and notes can be emphasized by resizing them or by colour coding them to highlight common group themes.

Arranging and emphasizing groups within the workspace. As mentioned in §8.3.2, groups can be moved and resized to affect their visual prominence. Because this usually runs into spatial constraints, people can also pan the workspace and zoom into areas to see more details. To view the contents of a single group in high detail one can select the group and say "zoom this" to see a smooth animation enlarging the group to fit the entire screen. To restore the view so that all notes are visible on the table users can use a "zoom all" speech command. Thus one can zoom in to inspect a group in detail, and then zoom out to move it to other related groups. I should mention that all workspace pan and zoom actions are global, and thus affect the entire table area.

Creating explicit meta-groups. Explicit meta-groups can be created in much the same way as groups are created. That is, people can create meta-groups by using two hand sides around existing groups (Figure 8.7, left) or they can lasso around existing groups using a single finger (right). Meta-groups behave as regular groups and can be easily moved with a single finger on an empty area or with five fingers over the entire group.

Linking notes and groups. Notes and groups can be linked using the annotation functionality of the Designers' Environment. A person says "annotation mode", and then draws lines linking related items; these lines will dynamically follow the notes and groups they are connected to. A person can also use a special multimodal command *"[point to note/group]* link this *[point to another note/group]* to this" where the note is selected using a single finger (as illustrated in Figure 8.7, left). A directional arrow is drawn on the destination node or group to reinforce the order of the hierarchy.

Colour coding groups. Finally, groups with similar themes can share a common colour. By default, groups are given the default color of the individual that created it. To modify the colour of a particular group, a person selects it with a single finger and says "make this <*say a colour*>" (Figure 8.7, right).

8.4 Design Issues

I developed the Designers' Environment to help us understand the design space of collaboration-aware multi-user multimodal interaction groupware for co-located interaction. Along the way, I encountered a number of design issues that should generalize to other groupware applications of this type. In this section, I detail these issues to provide insights for future designers.

8.4.1 Parallel Work

As mentioned in the introduction, the primary benefit of a collaboration-aware multi-user multimodal groupware system is that they recognize that multiple people are interacting with it, and they can allow co-located people to *simultaneously* gesture and speak commands to the groupware application. However, I found that the design of multimodal commands can greatly influence collaborators' propensity to simultaneously interact with the system. Some of the factors that affect parallel work include the effect of using of voice commands, the work area size, and the gesture size.

The effect of voice commands. The computer is able to isolate and recognize the speech of multiple people because each person has their own microphone connected to separate speech recognizers. This means that multiple people can simultaneously issue voice commands to the system. The problem is that voice commands are also a very public action: all voice commands are audible by others in the collaborative process regardless of their current activity. Yet in practice social protocols discourage collaborators from speaking simultaneously over each other. Thus while the system allows parallel activity, people may choose to work sequentially instead. To mitigate this effect, I argue that designers of collaboration-aware multi user multimodal systems should avoid using voice commands for those actions that are likely to be done simultaneously by multiple people, but that they should use voice commands when people are likely to interleave their utterances [Tse *et al.*, 2007a]. For example, the KJ Method encourages people to jot down notes simultaneously. While I could have used voice recognition for rapid entry of notes, this may have inhibited simultaneous entry. Instead, I chose to let people enter notes using a pen on the tablet as this method lets people easily engage in parallel individual work.

The effect of small work areas and large gestures on individual work. People often work on highly individual tasks, even when working together towards a common goal. A gestural interface, especially one that requires large gestures over a small table, may affect people's ability to do their individual work in parallel with others. Other's gestures may be distracting or simply get in the way. On large tables people create personal work areas to make it easier to pursue individual work [Hall, 1966, Rogers and Lindley, 2004, Scott *et al.*, 2004]. In this case, the table was somewhat small. Thus I gave people individual Tablet PCs that serve as personal work areas, where people used it to create and publish notes. More generally, I argue that designers can increase the amount of parallel work by ensuring that each collaborator has enough space to serve as a personal work area. This could be part of the table, or it could be a separate device as I have done.

The effect of gesture size. Generally speaking, manipulating artefacts on larger work surfaces require larger gestures, e.g., as people move items, or when they group a large set of notes. This is beneficial as large gestures are more visible to other collaborators; they involve more motion and often involve the movement of the entire arm. This produces consequential communication that others can use for awareness and coordination [Gutwin and Greenberg, 2004]. However within the context of parallel work, this awareness can distract from individual concentration. Large gestures – in addition to requiring more time to complete – can shift the attention of collaborators away from their current task. Conversely a smaller gesture could result in actions that are less visible to others. A balance must be struck between the two. Designers wishing to increase parallel individual work could benefit from using smaller gestures. Conversely, if designers wish to increase collaboration and communication, they could benefit from using larger gestures For example, consider the difference between the two methods of creating labels in the Designers' Environment. The annotation method encourages collaboration, as one has to write in large letters directly atop the table. The tablet method encourages individual work, as it uses a small gesture area for writing on the tablet that is generally hidden from others. As another example, the large five finger gesture for moving groups is more visible than the one-finger gesture for moving a single note. This makes sense, as the group movement of Step 4 should inspire more conversation than note placement of Step 2 (Figure 8.2).

8.4.2 Mode Switching

Since collaboration-aware multi-user multimodal systems recognize multiple people, each person can potentially be working and switching between separate individual modes. For example, one person could be in an 'annotation mode' while another is in a 'moving mode'. While seemingly beneficial, I encountered three key issues that made individual modes difficult for multiple people to understand; these are described below.

Public voice commands. Some voice commands trigger individual mode switching, while others could trigger global mode switching. The problem is that one person's voice command is heard by others, and this may mistakenly give others the false impression that they are in the same mode. For example, they may believe that the mode switch is global when it is in fact individual, or that the mode switch does not affect them when it is in fact global. One solution uses clearly stated voice commands that result in global mode switches (e.g., the 'Annotation Mode' or 'Zoom this' speech command), while favouring gestures for commands that result in individual mode switches (e.g., moving and/or scaling a note).

Mode Visualization. If modes cannot be avoided in the design of a collaboration-aware multi-user multimodal system, some mode awareness is crucial to avoiding confusion regarding individual modes. This is usually done by altering the appearance of the objects affected. In distributed groupware, modes are often suggested by overlaying mode information on the telepointer. Yet in direct touch environments such as a digital table – where a pen or finger directly interacts with the digital display – information in the immediate vicinity of the touch point can be occluded by a person's hands. One partial solution slightly offsets the mode visualization so that it is not occluded by the hand [Vogel and Baudisch, 2007]. Another solution could highlight what objects or areas are affected by a user's touch. For example, we already saw how a yellow border around the table's perimeter is used to mark the global annotation mode (Figure 8.6, left).

Global action awareness and interruption. Global actions on the table, while often necessary, can be problematic. They can be extremely distracting to others, or can lead to changes that others do not want. I suggest two ways of mitigating these problems. The first method is to increase people's awareness of another's global acts. I do this by leveraging public speech commands and by using large gestures. Both produce consequential communication that others can use for coordination [Gutwin and Greenberg, 2004]. *Continuous* global workspace manipulations leverage large whole handed gestures, and take time to do. *Discrete* commands unfold over time rather than done all at once by invoking an animation sequence so the action takes time to unfold. This increases their visibility. Second, I allow others to interrupt a global action when they do not agree with it or if they want to discuss it further. They can stop a person in the middle of a continuous manipulation, or touch the digital table to stop the animation of a discrete manipulation.

Artefact modes. The behavioural characteristics of an artefact should try to match people's expectations. Confusion can arise when people expect modes to extend only to the artefact. For example, people using the Designers' Environment expected to be able to write annotations on groups and have them move with the group. Instead, the system provided a global "Annotation Mode" where annotations would reside in a layer above other artefacts (such as notes and groups). Thus, annotations would stay in place even if a group was moved. An alternative may have been to provide a gesture or toggle switch to allow a single finger to annotate over a group.

In summary, we already know that modes in traditional interfaces should be avoided, although this is hard to do in practice. The same is true in multi-user multimodal systems. Designers should try to avoid introducing modes if they can. If modes are necessary, they should carefully consider people's mode expectations, how they understand each other's multimodal mode switching actions, and how modes are visualized on the surface.

8.4.3 Personal and Group Territories

People naturally establish personal and group territories in the collaborative workspace [Kruger *et al.*, 2004, Scott *et al.*, 2004]. Consequently, the design of items, item containers and interactions on the workspace can have a significant impact on how people establish these territories. In this section we take a closer look at artefacts and how they affect territories on the digital surface.

Items such as individual images, notes, or illustrations can be rotated to establish personal and group territories. Studies have shown that in collaborative work, people often rotate items towards themselves in their own personal territories, while items inside a group territory are typically rotated in a compromised orientation that all collaborators can read [Kruger *et al.*, 2004]. In some cases, the designer may want to promote the table as a group territory by hindering the establishment of personal item territories. This could be done by limiting the rotation of artefacts to a single orientation. For example, since the goal of the KJ method is to treat each idea as equal, collaborators sit along a single table side and organize notes using a common orientation (Figure 8.2). This practice is replicated in the Designers' Environment.

Item size on a digital table also has an impact, where its visual prominence influences the amount of attention it receives from collaborators [Scott *et al.*, 2004]. For example, since there was limited screen real estate in my implementation, individual notes were cropped to remove any blank space not occupied by ink strokes. Thus each note would consume a minimal amount of screen space on the digital table. In practice however, this made notes with lots of text more visually prominent. Ideas would have been treated more equally if all notes were the same size as they are in the KJ Method.

Item containers are areas of the digital surface used to hold items, and can include tools for sorting, labelling, organizing and relating the contained items. To encourage discussion, I identified the person who made the group by coloring the container with that person's colour. Yet this can mistakenly give the impression that these containers are a personal territory, and that others should not manipulate its contents. In practice, container marking and location on the table can profoundly affect how they are viewed as personal vs. group territories.

8.4.4 Joint Multimodal Commands

Joint multimodal commands are commands issued by multiple people that overlap (or interleave) with one another in time. There are two types of joint commands: Independent joint commands and dependent joint commands.

Independent joint commands happen when people interact simultaneously (but independently) to achieve a joint action that might otherwise require several sequential steps by one person working alone. For example, in the Designers' Environment it is possible to move groups and to pan the workspace at the same time. Two people can work together to move a group to an off-screen location, e.g., one person could pan the workspace to the left to reveal an unused area as the other person moves the group to that spot. The result is that two people can move a group faster than a single individual could on their own.

For independent joint multimodal commands to work, people have to time and coordinate their actions closely. For this to work, the system must be very responsive. It has to animate changes in what seems like real time (so people get appropriate feedthrough of the other person's actions and resulting state). It must also carry out commands with no delay. Using the example above, if a person tries to drop a group as the other is panning, lags in either the panning or in the drop action could result in the group being misplaced.

Dependent joint commands explicitly leverage the (speech and gesture) inputs of multiple people as a single command to the system. For example, Figure 1 shows an instance of joint grouping in the Designers' Environment. Two people are selecting notes simultaneously: the woman uses the command "select this *[note]*" while the man uses a hand bracketing gesture and says "group selected items." The end result is that the items selected by the man with the hand bracketing gesture and the single item selected by the woman will be combined into a single group. Under the covers, the interleaving multimodal grouping command searches for selection actions made by other collaborators within a 5 second threshold and adds them to the current grouping command.

When dependent joint commands are used, it can be difficult to achieve appropriate timing for closure. Usually joint multimodal commands accept input from collaborators within a time window of several seconds before and after the joint command has been issued. A *short joint command window* could result in collaborator's inputs being missed in the joint command. Consider Figure 1, if the man said "group selected items" before the woman had finished adding the last item to the group a new group would be created with an item missing, and the woman's selection would still be active. To correct this action, one would need to undo the selection and move the note to the newly created group. A *long joint command window* could result in a collaborator's input being included by mistake. If the woman in Figure 1 wanted to create a separate group with her multimodal selection this input might be erroneously included into the man's multimodal grouping command. To correct this action one would need to find the note within the newly created group, remove it and reselect it so that it can be added to a new group. The timing of such joint multimodal commands will vary depending on the nature of the commands and the kinds of interactions seen in practice.

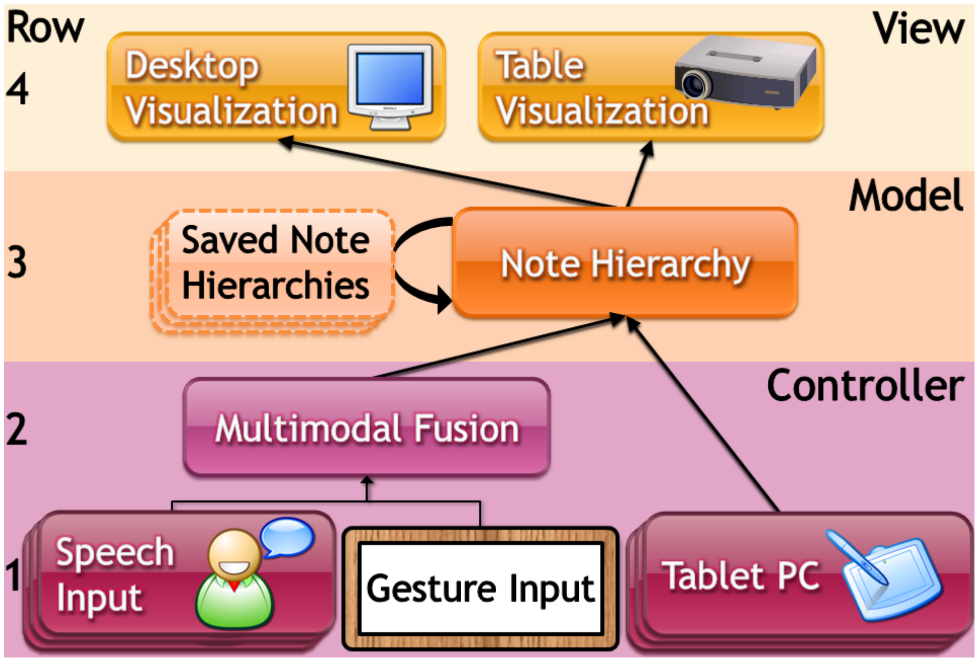


Figure 8.8. The Designers' Environment architecture

8.5 Architecture

While my primary goal is to consider design issues in collaboration-aware multi-user multimodal groupware systems, I also explain how I implemented this system for the benefit of future system designers.

A common approach used to support multiple people simultaneously interacting over artefacts in a shared workspace in distributed systems is to use a model-view-controller (MVC) architecture. Items on the shared workspace are stored on a single machine (model) and can be manipulated by multiple people working over separate computers (controllers) and they can view the shared workspace on their own screens (view). As illustrated in Figure 8.8, I leverage the model-view-controller approach to support multiple co-located people using speech and gestures in the Designers' Environment. All of the individual notes and their respective groups/links are stored in a dictionary of key/value pairs: this is the model (Figure 8.8, row 3). Notes can be created using multiple Tablet PCs and manipulated using gesture and speech input, these are the controllers (Figure 8.8, rows 1&2). Collaborators can collaborate over a shared view on the digital table or they can view notes and manipulate notes after the meeting on a standard desktop computer, this is the view (Figure 8.8, row 4).

8.5.1 Model

To allow simultaneous input from multiple people in the co-located setting we first need a common model for manipulating data across multiple input devices and computers. I store all note hierarchy information in a dictionary of key-value pairs using a distributed networking toolkit (the GroupLab Collabrary [Boyle and Greenberg, 2005]). This networking toolkit stores all note information on a common shared sever. It allows distributed and local controller applications to modify values within the dictionary and provides notifications of updated values to subscribed view applications. As illustrated below, I store each note and image as a separate key value pair, the key represents the note/image number and the value contains the ink strokes/image data serialized to a byte stream. The relations between notes are stored in the dictionary as a separate key representing an ID and a value containing two object IDs (notes, groups, images, etc) for starting and ending nodes. Groups contain a list of items, item positions and region information describing the bounding region and lasso points.

```
/note/1  = [ Ink Strokes ]
/image/2 = [ DesignSketch.jpg ]
/link/3  = /note/16, /image/7
/group/1/items = /note/1, /image/2, /group/4
/group/1/itemPositions = (X1,Y1), (X2,Y2), (X4,Y4)
/group/1/Region = (X, Y, Width, Height, Lasso)
```

8.5.2 Controllers

To support public and private actions in a multimodal co-located environment we need to provide collaborators with a variety of interaction options (as illustrated in Figure 8, row 1).

Handwriting recognition opens up new interaction possibilities for those used to pen and paper as it is possible for people to leverage more capabilities of computers. I perform handwriting recognition using the Microsoft Tablet PC SDK to convert each written note to text form. The recognition results are used in speech selections (e.g., "select tinted windows"), note conversion (e.g., "recognize this *[note]*") and web search queries on the Tablet PC or Table (e.g., "*[point to note]* find related images"). The most likely result is placed as a search query (to Google Images) into a browser window where the user can then click on links and images as desired. On the Tablet PC when a user clicks on the Send to Table button or when a user says "convert to note" on the digital table, the system checks if the current web address is an image. If it is an image, it sends that image to the server. Otherwise it uses the GroupLab Collabrary [Boyle and Greenberg, 2005] to capture an image of that web page and sends the image to the server. As well, the ink strokes of a note are serialized for transportation using the Tablet PC SDK and sent to the server.

Speech Commands. The speech controller is tightly coupled to the contents of the model, where the speech vocabulary is extended on the fly to include the recognized text of new notes. For example, if someone uses the Tablet PC to add a note with "halogen headlights" as the contents, this will appear in the model and the speech command vocabulary will be expanded to include a "select halogen headlights" command. In my implementation, speech commands are recognized from multiple people using noise cancelling Labtec LVA 7330 headsets connected to individual Tablet PCs. It is also possible to recognize the speech of multiple people using multiple sound cards. From the software perspective, I use GSI DEMO [Tse *et al.*, 2006c] a toolkit for gesture and speech interaction in co-located environments to send the speech recognition results from multiple tablet PCs to the server hosting the note hierarchy (the model).

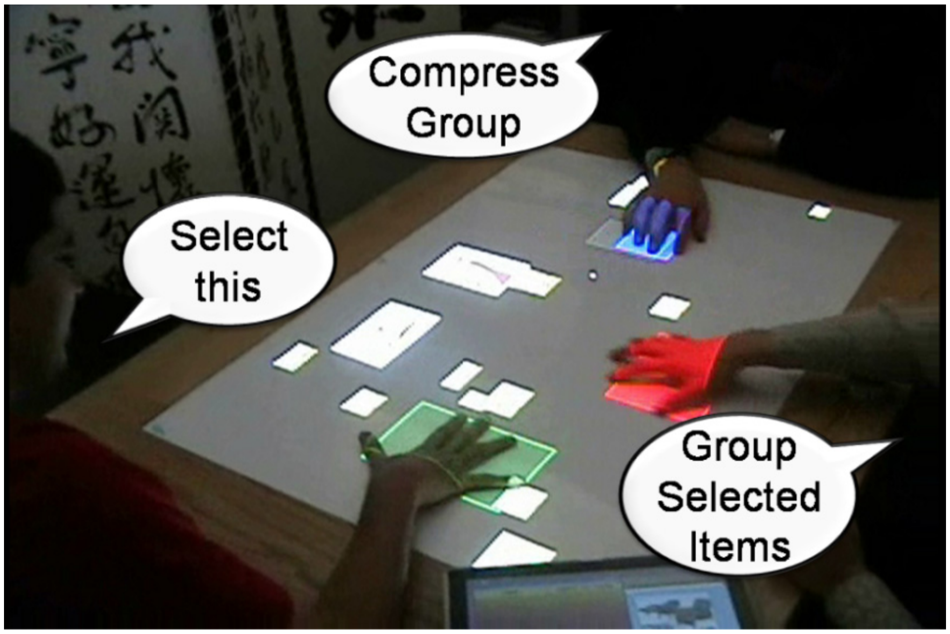


Figure 8.9. The Double Diamond Touch showing multi-user identification

Gesture Commands. Three basic functions are needed to support simultaneous artefact manipulation by multiple people on a digital table display. First, multiple simultaneous inputs (at least one per user) need to be detected by the digital table. Second as discussed in §8.4.1, the digital table must provide a reasonable amount of space for people to engage in parallel work. Finally, the digital table must be able to determine the person generating the touch if user identity is to be used in the application (e.g., for the multi-user multimodal fusion described next). In my implementation I originally chose to use a Diamond Touch table to detect multiple simultaneous inputs from multiple people. However, I wanted a larger table size so that three or four people could have their own non overlapping personal space for object manipulation. For this I chose a 148 x 116 cm Double Diamond Touch digital table [Tse *et al.*, 2007b] as illustrated in Figure 8.9. This one-off table comprises two standard Diamond Touch tables, where it is treated as a contiguous surface. Finally, the digital table also provides user level identification, as illustrated by the different coloured boxes in Figure 8.9.

Multimodal Fusion. The speech and gesture actions of a single person or multiple people can be fused into a single command (Figure 8.8, row 2). The Designers' Environment provides two types of multimodal command fusions: multimodal command unions and aggregate multimodal commands.

Multimodal command unions combine only a single speech and gesture input into a command e.g., "select this *[note]*". Multimodal command unions must include all appropriate speech and gesture components before it will be fused and passed on as a single command. If either the speech or gesture components are missing for a multimodal command union, the actions are ignored. For example, if the system recognizes a "select this" speech command and no one is pointing to a note then the command will be ignored. I extend the multimodal command unions presented by Cohen *et al.* [1997] to a multi-user setting by allowing others to include their speech and gesture commands as input to a multimodal command union. As discussed in §8.4.4, this can result in recognition errors if people are engaged in parallel work and do not intend to complete the multimodal command of their partner. For example, a partner may be moving a note at around the same time another person says "select this". I mitigate this issue in two ways: first, I allow others to complete a multimodal command only if the originator of the command does not complete it within a reasonable amount of time. Second, I only allow others to complete a multimodal command after it has been made public through a speech command, thus any prior artefact manipulations are treated as parallel work.

Aggregate multimodal commands can accept multiple speech and gesture inputs of multiple people, e.g. Figure 8.1 shows one person saying "*[point to note]* select this" while another says "group selected items". The issue of parallel work is exacerbated in the aggregate setting because prior commands also need to be considered as input. For example, the prior selections in Figure 8.1 could be used as input to the grouping command. My approach is to have a short (and customizable) time frame where the input of others can be included, also discussed in §8.4.4.

To implement multimodal fusion I used GSI DEMO [Tse *et al.*, 2006c] to collect the speech and gesture actions of multiple people into a single computer. I then fuse the speech and gesture actions of multiple people (based on the rules described above) into commands.

Turn taking policies can be used to avoid conflicts when multiple people try to manipulate the same object simultaneously. For small items I used a first person wins turn taking policy. E.g., when multiple people try to move a note at the same time in the Designers' Environment, the first person to come in contact with the note must release it before others can manipulate it. This policy allows people to move objects around the display without the fear of others manipulating the object under them. Conversely, for global workspace manipulations (like panning described in §8.3.4) I used a last person wins turn taking policy. E.g., a panning action can be interrupted by placing five fingers on the table. In practice, most conflicts are resolved by social protocol; the turn taking policies are designed to merely to assist the social process already used over physical tables.

8.5.3 Views

The purpose of the view is to present a visualization of the model (in this case the note hierarchy) for co-located collaborators (Figure 8.8, row 4). A shared tabletop view is beneficial in the co-located environment because people can see the digital content and the body language of others at the same time. If a shared view is used it should provide sufficient resolution for multiple people to manipulate artefacts in parallel. As illustrated in

Figure 8.9, I use two adjacent 1024x768 LCD projectors aligned along the long edge produce a total resolution of 1536x1024 on the digital table. Both projectors are connected to the server and the two displays as treated as one large display. This resolution is sufficient for viewing and manipulating around 50 notes on the digital table but would be difficult to manage over 100 notes without scaling. The view also provides a seemingly infinite digital workspace so that people do not feel constrained by the resolution limitations of the digital table. I achieve this in the Designers' Environment by allowing people to zoom individual notes or the entire workspace (§8.3.4).

Animations. In the co-located environment, the view should provide smooth animations to avoid the jarring effects of artefacts disappearing from a users view unexpectedly. For example, the "group selected items" would be confusing to others who might be manipulating a selected note, thus I smoothly animate the movement all of the selected notes into a new compressed group. Similarly, global transitions such as the "zoom all" command smoothly animate the scaling of the entire workspace.

Desktop reviewing. After the steps of the KJ method have been completed in the co-located setting, collaborators may wish to review groupings and notes at a later time on their own desktops. To support this, I provide a "save note hierarchy" speech command that saves the current note hierarchy to a file (Figure 8.8, row 3). The model can be later loaded using the "load note hierarchy" speech command or using a keyboard on any desktop computer. Items can still be viewed and manipulated using the mouse if desired.

Piccolo Direct3D. In my implementation the view is achieved using the Piccolo Direct3D toolkit by Bederson *et al.* [2004]. This toolkit provides a high level software application programmer's interface that allows notes to be efficiently rendered using graphics hardware accelerated primitives and textures. This toolkit also provides tools to simplify the animation of notes within the digital workspace and provides camera panning and zooming tools that make it trivial to develop a compelling zoomable workspace. Piccolo Direct3D provides the tools needed to create a smooth, responsive, and visually appealing user experience with the Designer's Environment.

8.6 Related Work

Computer support for designers. There are many tools designed to support informal brainstorming by designers (e.g., Cognoter [Foster and Stefik, 1986], Smart Ideas [SmartTech.com], PReSS [Cox and Greenberg, 2000]). Most existing systems are designed for a single person working with a keyboard and a mouse for jotting down ideas, or for a distributed group to work together. For example, PReSS [Cox and Greenberg, 2000] is intended solely for real time distributed interaction. While Cognoter is intended for people located in a meeting room, people contribute ideas by typing them on individual computers which then appear on a large wall display [Foster and Stefik, 1986]. In studies of this system, users had a tendency to focus on their own display rather than looking at the shared large display [Tartar *et al.*, 1991]. Other brainstorming systems are more oriented to group decision support, and they typically demand a rigid and formal process that must be followed exactly. This has proven ineffective for the informal brainstorming and sharing of ideas used in the early stages of design [Tartar *et al.*, 1991, Klemmer *et al.*, 2001]. Indeed, Buxton argued that the informal nature of sketches is crucial to creative design practice [Buxton, 2007].

KJ method. The KJ method is an established design practice, and several research systems have been designed to support this practice digitally. GUNGEN by Yuizono *et al.* [1998] provided support for the KJ method in a distributed environment where distance separated collaborators could still engage in collaborative brainstorming using a keyboard and mouse interface. The implementation of this system is very similar to Cognoter [Foster and Stefik, 1986].

Hybrid physical and digital interfaces. Several researchers have explored the use of hybrid physical and digital interfaces to support design practice. The Designers Outpost by Klemmer *et al.* supported a mix of physical and digital interaction as designers' could write on individual sticky notes and then have a camera capture the notes as they were placed on an upright SmartBoard [Klemmer *et al.*, 2001]. Lucero and Martens [2005] extended this interaction for creating mood (or emotion) boards on a digital table by mounting a camera above the table. The camera could capture images, magazine articles, and other physical objects placed on the digital table by momentarily turning the screen a green colour and performing background subtraction on the image.

Multimodal co-located interaction. A handful of systems have also explored how multiple people can interact with speech and gestures, although these were done over existing single user applications rather than over collaboration-aware groupware systems. In Chapter 3 I explored multi-user interaction over geospatial applications such as Google Earth, Warcraft III and The Sims [Tse *et al.*, 2006a]. These systems could not support parallel work as they were fundamentally limited by the one user per computer assumption of current operating systems. To work around this limitation, I explored a split view setting where two computer displays would be projected onto a shared digital tabletop in Chapter 7. Collaborators could work in parallel because they were working on separate computers. However, they could not engage in joint multimodal commands and interactions across displays.

8.7 Conclusion

In this chapter I presented a case study that explored the design and implementation of a collaboration-aware multimodal co-located system. The Designers' Environment was created to support the real world brainstorming practices of industrial designers on a multi user multimodal digital table. Using this case study I explored issues that future designers of multimodal co-located systems should consider. These issues include: parallel work, mode switching, personal and group territories, and joint multimodal commands. I also described a model, view, controller architecture for managing the simultaneous speech, gesture and pen inputs of multiple people over multiple computers. My goal is that this chapter would contribute to improving the design of future multi-user multimodal systems.

As a system, the Designers' Environment shows promise. While I did not do a formal study, colleagues have presented the Designers' Environment to industrial designers. They commented that they enjoyed being able to touch the table and interact with notes. They reacted positively to the features of the Designers' Environment not available in their physical paper setting (e.g., sorting). The designers also provided useful suggestions for features that the system could provide, some which were incorporated in the version of the

Designers' Environment reported in this chapter. Future work includes continuing the participatory design process with our industrial designers, to include note hierarchy logging and playback, support for different file formats, the ability to easily include physical media such as magazines, and to evaluate the refined system in practice as designers use it to brainstorm actual product ideas.

As a case study I argue that the Designers' Environment helped reveal issues that are valuable to the design of collaboration-aware multi-user, multimodal tabletop systems. Of course, I have barely scratched the surface of such systems. From the multimodal co-located perspective, I would like to explore the use of open speech vocabularies for things like dictation for note writing, searching and web browsing. There are questions about how I would link distant groups so that they could collaborate using the Designers' Environment. Mixed Presence Groupware could be used for real time interaction of multiple groups of people interacting with the Designers' Environment [Tang *et al.*, 2006a]. While both multi-user interaction and multimodal interaction have a long history, the combination of the two is still fairly novel. I recognize that there is much left to do.

In the following chapter I summarize and reflect upon the contributions of this dissertation by concluding with a discussion of my thesis objectives.

Chapter 9. Conclusions

In this chapter I conclude this dissertation with a high level discussion of my contributions and future work. I begin with a discussion of how I have accomplished the thesis objectives set in Chapter 1. These accomplishments form the basis of the research contributions of my thesis. I then discuss what we have learned and conclude with a discussion of future research directions. This chapter is brief. The individual components of the thesis are not evaluated or criticized as this has been done at the end of each chapter.

9.1 Overall Contribution

As stated in Chapter 1, the over-arching goal of my research is to inform the design and development of technologies to support multimodal co-located collaboration where multiple people interact with speech and gestures over a digital table. The thesis is that such systems can be reasonably built, that they are beneficial, and that this new area of group interaction design is worthy of further research. To achieve this goal I performed a series of exploratory probes into multimodal co-located interaction. I showed that such systems can be constructed through programming by demonstration collaboration-transparent wrappers atop of existing single user applications and as collaboration-aware groupware systems. I showed that such systems are beneficial from both a theoretical and empirical perspective. And I explored how multi user multimodal interaction can be designed to support parallel work. Thus the overall contribution is to show that multimodal co-located interaction is indeed worthy as a new paradigm for tabletop interaction.

9.2 Research Contributions

The research contributions of this thesis are attained from the accomplishment of my thesis objectives. This thesis explores a breadth of different problems related to multimodal co-located interaction. I emphasize that the goal of this dissertation is to provide initial insights and explorations in multimodal co-located interaction to direct future research.

In this section, I summarize what has been learned about multimodal co-located interaction and I later describe how future research could leverage the lessons learned in this dissertation. The seven research contributions of this thesis are described below.

First Contribution. I distilled existing theories, empirical and ethnographic studies into a set of behavioural foundations to inform the design of multimodal co-located systems and list individual and group benefits.

We have improved our understanding of individual and group benefits of using speech and gesture commands in a multi user setting. In Chapter 3, I described a set of behavioural foundations motivating multimodal co-located interaction. I distilled a number of theories of team work, empirical and ethnographic research into a set of individual and group motivations for multimodal co-located interaction. These implications resulted in a section that discussed implications for design suggesting how these foundations could be applied to the design of interfaces that leverage the benefits of multimodal speech and gesture interaction in a co-located setting.

Second Contribution. I developed collaboration-transparent speech and gesture wrappers atop of existing commercial applications.

We have advanced our understanding of collaboration-transparent multimodal interface design for co-located systems over existing single user applications. To demonstrate how the behavioural foundations of my First Contribution could be applied to design, I designed three multimodal speech and gesture wrappers atop of existing geospatial applications for multi user interaction over a digital table in Chapter 3. I demonstrated how the expressive gesture and speech commands of prior ethnographic studies could be mimicked in my speech and gesture wrapper design. I discussed design issues encountered in the creation of these wrappers to inform designers of future multimodal co-located systems.

Third Contribution. I performed an observational study on how pairs used collaboration-transparent speech and gesture commands for collaboration with existing applications on a digital table.

We have furthered our understanding of how people use speech and gesture commands over existing single user applications on a digital table display. In Chapter 4, I observed how pairs used the speech and gesture wrappers designed in my Second Contribution for two collaborative tasks: trip planning and furniture layout. The results of this study revealed that people used speech and gesture commands as both commands to the computer and as awareness to other collaborators. I also observed that people interleave speech and gesture commands in both tightly and loosely coupled activity using the same input modality (e.g. speech) or different modalities.

Fourth Contribution. I developed GSI DEMO: a toolkit for rapidly prototyping multimodal co-located interactive systems

We now have tools to assist the implementation of systems that recognize the speech and gesture commands of multiple people and map that input into keyboard and mouse events over existing applications. GSI DEMO was meant allow multiple people to interact with speech and gestures over a digital table. As described in Chapter 5, GSI DEMO provides the following features:

1. The MERL Diamond Touch Gesture Engine recognizes different hand postures (e.g., arm, hand, five finger, fist, etc) and their respective dynamic movements (e.g., two fingers moving apart) for multiple people (up to four) on a digital table.
2. The GSI Gesture Speech Unifier fuses the speech and gesture commands of an individual or from multiple people into system commands.
3. GSI DEMO allows people to wrap existing single user application by demonstration, rather than programming. For example, continuous gestures are trained by saying "Computer, when I do [one finger gesture], you do [mouse drag]" (Figure 5.1).

This toolkit has also been extended to support the Double Diamond Touch described in Chapters 7 and 8. This was a custom hardware configuration designed specifically for supporting the research in this dissertation.

Fifth Contribution. I compared the performance of using speech for filtering selections over a large digital wall display to two commonly used gesture-only selection techniques.

We have insights into the performance cost of using speech and gesture commands for selection over a large digital wall display. In Chapter 6, I examined the performance of using speech to filter a dense selection space on a large wall display. Using an empirical pointing experiment comparing the speech-filtered bubble ray to two gesture only pointing standards, I discovered that multimodal selection was the fastest, least error prone and most preferred technique. These results suggest that the benefits of multimodal selection can outweigh the cognitive cost and physical cost of filtering the selection space with speech. While this study was performed for a single individual it's results have implications for the multi user setting as well since selection is a basic task that is heavily used in collaborative systems.

Sixth Contribution. I developed a system to support parallel work over existing applications on multimodal split view tabletop. I described three seating arrangements and three software configurations for a split view tabletop.

We have seen several approaches for supporting parallel work over collaboration-transparent wrappers over existing single user applications in a shared display multimodal environment. In Chapter 7, I explored the concept of multimodal split view tabletop interaction. A split view tabletop is a tabletop surface whose surface is split into two adjacent projected views. By connecting separate computers to each view multiple people can work simultaneously over each computer. Multimodal speech and gesture commands are used to augment awareness of collaborator's actions in the collaborative setting. I described different seating arrangements and software configurations for a multimodal split view tabletop. I also demonstrated multimodal split view tabletop interaction in practice through three case studies over existing applications.

Seventh Contribution. I developed the Designers' Environment: a collaboration-aware multimodal co-located system built from the ground up. Using this system, I discussed issues that future designers of true multimodal co-located systems should consider.

We have an understanding of how multimodal co-located interaction can be applied to a true groupware system. We also saw several design issues that arose in the true multimodal co-located setting. In Chapter 8, I discussed the development of a system to support the initial brainstorming creativity practices of industrial designers. Using the four steps of the KJ creativity method, I developed multimodal interaction techniques to complement the existing work practices of designers. During the development of this system I encountered a number of design issues that I detailed for the benefit of future multimodal co-located system designers. There design issues covered topics such as parallel work, mode switching, personal and group territories, and joint multimodal commands. I also described a model, view, controller architecture for creating true multimodal co-located applications.

9.3 Advancing our Understanding of Multimodal Co-located Interaction

As mentioned, this thesis explores the design and technical development of technologies that support multiple co-located people collaborating with multimodal speech and gesture commands over digital table. At the beginning of this thesis, we began with a list of questions that have now been initially explored in this thesis. In this section, I summarize these findings by describing what we have learned and how others can leverage this work. There are three main areas explored in this thesis.

First, we saw benefits for speech and gesture commands in a co-located setting through a set of individual and group benefits based on theories, empirical and ethnographic research on how people collaborate in everyday settings. We then explored multi user speech and gesture interfaces atop of existing single user applications. Through an observational study we learned how people use these wrappers for collaborative work and found that people overwhelmingly used speech and gestures as both commands to the computer and as communication to other collaborators.

We examined the effectiveness of speech filtering for selection over a large digital wall display. I performed a controlled evaluation and found that multimodal commands improved selection efficiency, accuracy and user preference over two gesture-only selection techniques.

The remainder of this thesis explored further technical explorations in the design of multimodal co-located applications. I explained how to create multimodal co-located systems using GSI DEMO to map speech and gesture commands to keyboard and mouse actions by demonstration rather than programming. For example, continuous gestures were trained by saying "Computer, when I do [one finger gesture], you do [mouse drag]".

We looked at parallel work over existing applications using a multimodal split view tabletop. By using separate computers we saw how pairs could work in parallel over existing single user application over a shared tabletop surface. Multimodal commands provided enhanced activity awareness especially when people were working in parallel.

Finally, we investigated how multimodal co-located interaction could be applied to a true groupware system in a system for supporting the brainstorming activities of industrial designers. We saw that several design issues that arose in the true multimodal co-located setting.

9.4 Reflections

Moving beyond the particulars of this thesis, we can reflect on how the general idea of multimodal co-located interaction informs Computer Supported Cooperative Work (CSCW).

In traditional co-located interaction over digital tables, the focus has been primarily on adapting existing Human Computer Interaction paradigms to a large touch screen. While powerful, co-located interaction can cover many more interaction modes than just touch. We already saw the interaction possibilities provided when multiple collaborators exploit speech and gesture modalities for interaction. Yet other modalities exist: eye-gaze, body postures, tangibles, and so on. There is no question that CSCW can enrich co-located interaction by considering these other modes as well.

Similarly while most co-located CSCW research has focused around interaction with a computer system, this thesis emphasizes the value of commands that also serve as communication to others. In the same vein, designers of future co-located collaborative systems should also consider the public nature of the commands/actions that they are creating. This is important, for the high level of awareness provided by consequential communication also provides opportunities for people to engage in actions across multiple collaborators.

This thesis also extends research in sharing single user applications to arenas that are beyond the desktop. In particular, it explored how existing single user applications could have completely remapped inputs to make them more suitable for keyboard and mouse interaction. Within CSCW, screen sharing was limited mostly to how people share applications on traditional mouse/keyboard workstations. From our new understanding, the design focus of sharing single user applications can be extended considerably in a co-located setting. First, sharing is more than just having access to input devices. Rather, it should enable publicly perceptible actions. Second, it should be easy to rapidly prototype these actions. We already saw how GSI Demo allowed people to map speech and gesture interface atop of existing applications so that they could focus on the higher level actions, rather than the lower level mouse and keyboard simulations. However, GSI Demo is a fairly limited system, and we can do much better than that (see 9.5) Third, CSCW has mostly considered turn taking policies in distributed groupware. It must also incorporate such policies in co-located settings in order to mitigate the effects of simultaneous interaction over not only single user applications, but multi-user ones as well [Scott *et al.* 2003].

9.5 Looking to the Future

The breadth of future research in multimodal co-located interaction is large as there remains much to be explored. Future work in this area can be divided into three threads: empirical, technical, and real work practice. Empirical future work will explore the nuances of using speech and gesture commands in a co-located setting to inform future technical implementations. Technical explorations involve further system implementations and interaction techniques for multimodal co-located interaction. Finally real work practice involves applying both empirical investigation and technical implementation to real world situations and environments. I describe each of these in turn.

Empirical future work. Earlier, in Chapter 3 I reviewed existing theories, empirical and ethnographic research on how people use speech and gestures from the perspective of multimodal co-located interaction. In Chapter 4, I observed that people used speech and gesture commands in very similar to the manner that existing empirical and ethnographic research predicts. Of course, I also noticed that people interleaved speech and gesture commands across the pair in ways I had not originally anticipated. These interleaving acts could form the basis of future investigations in understanding the nuances of multimodal co-located interaction. Similarly, my initial investigations into true multimodal co-located interactions (in Chapter 8) revealed a number of design issues that could be used as topics for future empirical investigations.

Empirical investigation also needs to be pursued on different configurations to examine if the results of these studies generalize to other settings. While the empirical studies of this thesis were limited to pairs of collaborators, there is currently no reason to think that these results will not apply to groups of three and four. Future studies could investigate varying the number of collaborators (e.g. groups of three or more), the size and orientation of the display (e.g. horizontal versus vertical), the type of tasks (sequential versus parallel), the number of displays (e.g. smart rooms, multimodal split view tabletops), and the modalities used (e.g., gaze, speech, gesture). Further investigations would provide a greater understanding of the nuances of using multimodal commands in a co-located setting. For example, in true multimodal co-located systems (as in Chapter 8), it was noticed that the design of multimodal commands improved collaborator's awareness of each others activities as a cost of the amount of parallel work that a group might use. A study could be performed to compare multimodal commands to gesture-only interaction, measuring the awareness that people have of collaborators' activities and comparing that with the amount of parallel work they engage in.

Technical future work. There is much technical work that needs to be done before we can deploy multimodal co-located interaction in a practical setting. While the focus of my research has not been on improving gesture recognition and speech recognition technologies, advances in such technologies will greatly improve the performance and robustness of multimodal co-located interaction in practice. From the interactions perspective future technical explorations involve improving tools for developers of multimodal co-located systems and investigating new multi user multimodal interaction techniques.

In terms of building tools for developers, one could simplify the task of mapping a large number of multimodal functions to keyboard and mouse commands by improving the generalization capabilities of GSI DEMO. That is, if GSI DEMO could understand that a person was trying to map the "fly to <place>" command for many different cities, it could present a simple interface for adding new place names. GSI DEMO could leverage programming by demonstration technologies to further improve the efficiency of creating large amounts of multimodal mappings. Also, by detecting dynamic hand posture movements, GSI DEMO could be expanded to enable gestures to map onto keyboard and mouse event sequences. For example, a whole hand erasing gesture over a folder window could erase all the items in a window, or a two finger gesture over a file folder could open or close that folder.

From the software development perspective, GSI DEMO provided facilities for accessing input from multiple people but it does not provide any graphics support. While existing toolkits such as Piccolo [Bederson *et al.*, 2004] provide hardware optimized graphics support for handling graphics, developers still need to allow graphical widgets to understand concurrent multimodal input. A widget layer on GSI DEMO would greatly simplify the task of developing custom graphical widgets without the hassle of sending speech and gesture event information to each widget manually.

With the proper software development tools it will be possible to explore new interaction techniques that examine the capabilities and limitations of using multimodal speech and gesture commands in a co-located setting. New interaction techniques are important because the majority of interactions with personal computers are designed specifically for a keyboard and a mouse. Since speech and gestures have very different affordances than a keyboard and a mouse, they require appropriate interaction techniques. One such distinction is that multimodal commands blur the distinction between commands to a computer and conversation with other collaborators. Since computers lack many of the visual and verbal cues that people naturally use to establish if someone is talking to them they could mistakenly interpret conversation with others as commands to the computer. Future interaction techniques could explore ways that people could signal their intentions to the computer. For instance studies have shown that people naturally increase the volume of their speech when issuing commands to the computer [Lunsford *et al.*, 2005]. This cue could be used to increase the likelihood that a spoken phrase was intended as a command to the computer.

Real work practice. The original motivations for this work stems from an understanding of people's collaborative work practices in real world situations. Since multimodal co-located interaction investigates people's natural collaborative work practices, an understanding of how multimodal co-located systems might reveal issues that would otherwise be missed in a laboratory setting. While the large scale deployment of multimodal co-located systems seems unpractical, researchers can leverage the domain knowledge of experts to design prototype systems for current practice. These systems can be evaluated offline by inviting practitioners to engage in simulated tasks that are representative of real collaborative situations. For example, this process was partially used in the development of the Designers' Environment, as industrial designers took part in the participatory design process and the system was later presented to them for evaluation. A continuation of this process would yield much more viable results. Of course, a similar approach could be applied many other domains such as military command and control, air traffic control, and surgical planning.

Given the high cost of deploying such systems, multimodal co-located interaction needs to prove its worth in practice before it will see adoption by numerous organizations. Based on the feedback that I have received from presentations of this work, there is significant interest from the domain of safety critical applications in publicly perceptible interactions that can be used in a face to face setting. As the prices of large digital displays continues to decline, I expect that this research will also be of interest to the general public as people move progressive towards using digital content in face to face meetings. The goal of this dissertation is that future designers will leverage the work described in this thesis to create appropriate technical solutions for cooperative work. A quote from the science fiction author William Gibson:

"The future is already here – it's just not uniformly distributed."

References

1. Agrawala, M., Beers, A. C., McDowall, I., Fröhlich, B., Bolas, M., and Hanrahan, P. (1997) The two-user Responsive Workbench: support for collaboration through individual views of a shared space. Proc. Computer Graphics and interactive Techniques, ACM Press, 327-332.

2. Aliakseyeu, D., Subramanian, S., Lucero, A., and Gutwin, C. (2006) Interacting with piles of artifacts on digital tables. Proc. AVI '06, ACM Press, 159-162.

3. Barthelmess, P., Kaiser, E., and McGee, D. R. (2007) Toward content-aware multimodal tagging of personal photo collections. Proc. ICMI '07, 122-125.

4. Baudel, T. and Beaudouin-Lafon, M. (1993) Charade: remote control of objects using free-hand gestures. Communications of the ACM, 36(7), July, 28-35.

5. Baudisch, P., Cutrell, E., Robbins, D., Czerwinski, M., Tandler, P., Bederson, B., and Zierlinger, A. (2003). Drag-and-Pop and drag-and-pick: Techniques for accessing remote screen control on touch and pen operated systems, Proc. Interact, 57-64.

6. Bederson, B. and Hourcade, J. (1999), Architecture and implementation of a Java package for Multiple Input Devices (MID), HCIL Technical Report No. 9908, http://www.cs.umd.edu.hcil.

7. Bederson, B., Hollan, J., Druin, A., Stewart, J., Rogers, D. and Proft, D. (1996), Local Tools: An Alternative to Tool Palettes, Proceedings of the ACM Conference on User Interface and Software Technologies (UIST '96), Seattle, pp.169-170.

8. Bederson, B., Grosjean, J., & Meyer, J. (2004). Toolkit Design for Interactive Structured Graphics, IEEE Transactions on Software Engineering, 30 (8), pp. 535-546.

9. Bekker, M.M., Olson, J.S., & Olson, G.M. (1995). Analysis of gesture in face-to-face design teams provides guidance for how to use groupware in design. In *Proceedings of the Symposium on Designing Interactive Systems 1995,* pp. 157-166.

10. Bentley, R., Hughes, J., Randall, D., Rodden, T., Sawyer, P., Shapiro, D. And Sommerville, I. (1992). Ethnographically-informed Systems Design for Air Traffic

Control. In *Proceedings of Computer-Supported Cooperative Work (CSCW) 1992*, pp. 123-129.|

11. Bezerianos, A., Balakrishnan, R. (2005) The Vacuum: Facilitating the manipulation of distant objects. Proc. CHI 2005, ACM Press, 361-370.

12. Bier, B. and Freeman, S. (1991), MMM: A user interface architecture for shared editors on a single screen, Proceedings of the ACM Conference on User Interface and Software Technologies (UIST '91), Hilton Head, pp. 79-86.

13. Billinghurst, M. (1998) Put that where? voice and gesture at the graphics interface. SIGGRAPH Computer Graphics 32, 4, 60-63.

14. Blanch, R., Guiard, Y., and Beaudoin-Lafon, M. (2004) Semantic pointing: improving target acquisition with control-display ratio adaptation. Proc. ACM CHI '04, 519-525.

15. Bolt, R.A., Put-that-there: Voice and gesture at the graphics interface. *Proc ACM Conf. Computer Graphics and Interactive Techniques Seattle*, 1980, 262-270.

16. Booth, K. S., Fisher, B. D., Lin, C. J., and Argue, R. (2002) The "mighty mouse" multi-screen collaboration tool. Proc. UIST '02. ACM Press, New York, NY, 209-212.

17. Boyle, M. and Greenberg, S. Rapidly Prototyping Multimedia Groupware. Proc Distributed Multimedia Systems (DMS'05), Knowledge Systems Institute, 2005.

18. Bricker, L., Baker, M., Fujioka, E. and Tanimoto, S. (1999), A System for Developing Software that Supports Synchronous Collaborative Activities, Proceedings of EdMedia, pp. 587-592.

19. Buxton, W.A.S. (1995) Chunking and phrasing and the design of human-computer dialogues. Human-Computer Interaction: Toward the Year 2000, Morgan Kaufmann Publishers Inc., pp. 494-499.

20. Buxton, W. (2007) Sketching User Experiences: Getting the Design Right and the Right Design, Morgan Kaufmann, ISBN-13: 978-0123740373.

21. Buxton, W., Fitzmaurice, G. Balakrishnan, R. & Kurtenbach, G. (2000). Large Displays in Automotive Design. IEEE Computer Graphics and Applications, 20(4), 68-75.

22. Cao, X. and Balakrishnan, R. Evaluation of an online adaptive gesture interface with command prediction. *Proc Graphics Interface*, 2005. 187-194.

23. Chin, T., Doctors Pull Plug on Paperless System. American Medical Association News, Feb 17, 2003. http://www.ama-assn.org/amednews/2003/02/17/

24. Clark, H. *Using language.* Cambridge Univ. Press, 1996.

25. Cohen, P., Johnston, M., McGee, D., Oviatt, S., Pittman, J., Smith, I., Chen, L., Clow, J. (1997) Quickset: Multimodal Interaction for Distributed Applications, Proceedings of Multimedia '97, pp. 31-40.

26. Cohen, P. Speech can't do everything: A case for multimodal systems. *Speech Technology Magazine*, 5(4), 2000.

27. Cohen, P.R., Coulston, R. and Krout, K., Multimodal interaction during multiparty dialogues: Initial results. *Proc IEEE Int'l Conf. Multimodal Interfaces*, 2002, 448-452.

28. Cohen, P. R. and McGee, D. R. (2004) Tangible multimodal interfaces for safety-critical applications. Communications of the ACM 47, 1 (Jan. 2004), 41-46.

29. Coleman, M. L. (1969) Text Editing on a Graphic Display Device Using Hand-Drawn Proofreader's Symbols, Proc. 2nd University of Illinois Conference on Computer Graphics.

30. Corradini, A., Wesson, R., Cohen, P. (2002) A Map-Based System Using Speech and 3D Gestures for Pervasive Computing. Proc. of IEEE International Conference on Multimodal Interfaces, IEEE Computer Society, 191-196.

31. Cox, D. and Greenberg, S. (2000) Supporting Collaborative Interpretation in Distributed Groupware. Proc. CSCW 00, ACM Press, 289-298.

32. Cypher, A. *Watch What I Do: Programming by Demonstration*. MIT Press, 1993.

33. Davis, K., Biddulph, R., Balashek, S. (1952) Automatic Recognition of Spoken Digits. The Journal of the Acoustic Society of America, 24(6), 637-642.

34. Dietz, P.H., Leigh, D.L., DiamondTouch: A Multi-User Touch Technology, *Proc ACM UIST,* 2001. 219-226

35. Dix, Alan; Finlay, Janet; Abowd, Gregory; Beale, Russel. (1998) Human Computer Interaction, Second Edition. Chapter 13,14. Prentice Hall International.

36. Dragicevic, P. and Fekete, J. (2004) The Input Configurator toolkit: towards high input adaptability in interactive applications. Proc. AVI '04. ACM Press, 244-247.

37. Duda, R., Hart, P., Stork, D. (2000) Pattern Classification, ISBN-10: 0-471-05669-3, ISBN-13: 978-0-471-05669-0 - John Wiley & Sons, Chapter 2. Maximum-Likelihood and Bayesian Parameter Estimation., pp 31, 50, 88.

38. Dudley, H. (1939) The Vocoder, Bell Labs Record, Vol 17, 122-126.

39. Elwart-Keys, M., Halonen, D., Horton, M., Kass, R., and Scott, P. (1990) User interface requirements for face to face groupware. Proc. CHI '90. ACM Press, 295-301.

40. Foster, G. and Stefik, M. (1986) Cognoter: theory and practice of a collaborative tool. Proc. CSCW '86, ACM Press, 7-15.

41. Fitts, P. M. (1954). The information capacity of the human motor system in controlling the amplitude of movement. Journal of Experimental Psychology, 47, 181-196.

42. Greenberg, S. (1990). Sharing views and interactions with single-user applications. Proc ACM/IEEE Conference on Office Information Systems, 227-237.

43. Greenberg, S., The Computer User as Toolsmith: The Use, Reuse, and Organization of Computer-based Tools. Cambridge University Press, 1993.

44. Greenberg, S. and Boyle, M. Customizable physical interfaces for interacting with conventional applications**.** Proc. ACM UIST Conference, 2002*,* 31-40.

45. Greenberg, S., Fitchett, C. (2001) Phidgets: Easy Development of Physical Interfaces through Physical Widgets. Proc. UIST '01, ACM Press, 209-218.

46. Greenberg, S., Gutwin, C., and Roseman, M. (1996). Semantic Telepointers for Groupware. Proc OzCHI '96, IEEE Computer Society Press. 54-61.

47. Grossman, T., Balakrishnan, R. (2005) The Bubble Cursor: Enhancing target acquisition by dynamic resizing of the cursor's activation area. Proc. CHI '05, 281-290.

48. Gutwin, C., and Greenberg, S. The importance of awareness for team cognition in distributed collaboration. In E. Salas, S. Fiore (Eds) *Team Cognition: Understanding the Factors that Drive Process and Performance,* APA Press, 2004, 177-201.

49. Gutwin, C., & Greenberg, S. (2000). The Mechanics of Collaboration: Developing Low Cost Usability Evaluation Methods for Shared Workspaces. IEEE 9th International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'00). June 14-16, held at NIST,Gaithersburg, MD USA.

50. Gutwin, C. and Greenberg, S. Design for individuals, design for groups: Tradeoffs between power and workspace awareness. *Proc. ACM CSCW,* 1998, 207-216

51. Hall, E. (1966) The Hidden Dimension. Anchor Books

52. Haller, M., Billinghurst, M., Leithinger, J., Leitner, D., and Seifried, T. (2005) Coeno: enhancing face-to-face collaboration. Proc. ICAT '05, vol. 157. ACM Press, 40-47.

53. Han, J. Y. (2005) Low-cost multi-touch sensing through frustrated total internal reflection. Proc. ACM UIST '05. ACM Press, 115-118.

54. Hancock, M., Carpendale, S. (2006) The Complexities of Computer-Supported Collaboration. Technical Report 2006-812-05, University of Calgary, Alberta Canada.

55. Heath, C.C. and Luff, P. Collaborative activity and technological design: Task coordination in London Underground control rooms. *Proc ECSCW*, 1991, 65-80

56. Hinckley, K. (2003) Synchronous Gestures for Multiple Users and Computers, Proc. ACM UIST '03.

57. Hinrichs, U., Carpendale, S., Scott, S. (2005) Interface Current Supporting Fluent Collaboration on Tabletop Displays. Proceedings of Smart Graphics 2005.

58. Hollan, J., Hutchins, E., & Kirsh, D. Distributed Cognition: Toward a New Foundation for Human Computer Interaction. Proceedings of ACM TOCHI Vol 7 No 2 Jun 2000 pp. 174-196

59. Hutchins, E., (1995). Cognition in the Wild. MIT Press, Cambridge, MA.

60. Inkpen, K., Hawkey, K., Kellar, M., Mandryk, R., Parker, K.,Reilly, D., Scott, S., & Whalen, T. (2005). Exploring Display Factors that Influence Co-Located Collaboration: Angle, Size, Number, and User Arrangement. In Proceedings of HCI International 2005, July 22-27, 2005, Las Vegas, NV.

61. Isenberg, T., Miede, A., Carpendale, S. (2006) A Buffer Framework for Supporting Responsive Interaction in Information Visualization Interfaces. Proc. Conference on Creating, Connecting and Collaborating through Computing (C5 2006), IEEE Computer Society, pp. 262-269.

62. Ishii, H. and Ullmer, B. (1997) Tangible bits: towards seamless interfaces between people, bits and atoms. Proc. CHI '97. ACM Press, 234-241.

63. Johnston, M., Cohen, P., McGee, D., Oviatt, S., Pittman, J., and Smith, I. (1997) "Unification-based multimodal integration," in Proceedings of 35th Annual Meeting of the Association of Computational Linguistics, San Francisco, CA , 1997, pp. 281–288.

64. Jordà, S., Geiger, G., Alonso, M., and Kaltenbrunner, M. (2007) The reacTable: exploring the synergy between live music performance and tabletop tangible interfaces. Proc. TEI '07. ACM Press, 139-146.

65. Kabbash, P. and Buxton, W. (1995) The "Prince" technique: Fitts law and selection using area cursors. Proc. ACM CHI '95, 273-279.

66. Kaiser, E., Olvar, A., McGee, D., Benko, H., Corradini, A., Li, X., Cohen, P., Feiner, S. (2003) Mutual Disambiguation of 3D Multimodal Interaction in Augmented and Virtual Reality, Proceedings of ICMI-PUI 2003, 12-19.

67. Kaltenbrunner, M. and Bencina, R. (2007) reacTIVision: a computer-vision framework for table-based tangible interaction. Proc. TEI '07. ACM Press, 69-74.

68. Kilpatrick, P. J. (1976) The Use of a Kinematic Supplement in an Interactive Graphics System. Doctoral dissertation, University of North Carolina.

69. Klemmer, S. R., Newman, M. W., Farrell, R., Bilezikjian, M., and Landay, J. A. (2001) The designers' outpost: a tangible interface for collaborative web site. Proc. UIST '01, ACM Press, 1-10.

70. Krueger, M.W. (1977) Responsive Environments, Proc. American Federation of Information Processing Societies (AFIPS '77), Vol 46, 423-429.

71. Krueger, W., Bohn, C., Frohlich, B., Scheuth, H., Strauss, W., Wesche, G., (1995) The Responsive Workbench. IEEE Computer, 28 (7), 42-48.

72. Krueger, M. W. (1991) Artificial Reality II, Addison-Wesley Publishing Company,ISBN 0-201-52260-8.

73. Kruger, R., Carpendale, M.S.T., Scott, S.D., Greenberg, S. (2004) Roles of Orientation in Tabletop Collaboration: Comprehension, Coordination and Communication. In Journal of Computer Supported Collaborative Work, 13(5-6), 2004, pp. 501–537.

74. Latulipe, C., Bell, I., Clarke, C. L., and Kaplan, C. S. (2006) symTone: two-handed manipulation of tone reproduction curves. Proc. Graphics interface 2006, vol. 137, ACM Press, pp 9-16.

75. Lauwers, J. C., Lantz, K. A. (1990) Collaboration awareness in support of collaboration transparency: requirements for the next generation of shared window systems. Proc. CHI '90. ACM Press, 303-311.

76. Lee, S., Buxton, W., and Smith, K. C. (1985) A multi-touch three dimensional touch-sensitive tablet. Proc. CHI '85. ACM Press, New York, NY, 21-25.

77. Li, D. and Lu, J. (2006) A lightweight approach to transparent sharing of familiar single-user editors. Proc. CSCW '06. ACM Press, 139-148.

78. Lucero, A., Martens, J. (2005) Mood Boards: Industrial Designers' Perception of Using Mixed Reality. Proc. SIGCHI.NL Conference 2005, 13-16.

79. Lunsford, R., Oviatt, S., and Coulston, R., Audio-visual cues distinguishing self- from system-directed speech in younger and older adults. *Proc. ICMI,* 2005*,* 167-174.

80. MacKenzie, I.S. (1989). A note on the information theoretic basis for Fitts Law. Journal of Motor Behavior, 21:323–330.

81. Mackenzie, I. S. (1995) Movement time prediction in human-computer interfaces. In R. M. Baecker, W. A. S. Buxton, J. Grudin, and S. Greenberg, editors, *Readings in Human-Computer Interaction*. Kaufmann, second edition.

82. Malik,S., Ranjan, A., Balakrishnan, R. (2005) Interacting with large displays from a distance with vision-tracked multi-finger gestural input. Proc. UIST 2005, ACM Press, 43-52.

83. Magerkurth, C., Memisoglu, M., Engelke, T., and Streitz, N. (2004) Towards the next generation of tabletop gaming experiences. Proc. GI '04, vol. 62, 73-80.

84. Markowitz, J. (1996) Using Speech Recognition, Prentice Hall, ISBN 0-13-186321-5.

85. McGee, D., Cohen, P. (2001) Creating Tangible Interfaces by Augmenting Physical Objects with Multimodal Language. Proceedings of IUI '01. pp. 113-119

86. McGuffin, M., Balakrishnan, R. (2005) Fitts law and expanding targets: Experimental studies and designs for user interfaces. ACM TOCHI, 12(4), ACM Press, 388-422.

87. McNeill, D. 1992. Hand and Mind: What Gestures Reveal About Thought. University of Chicago Press, Chicago.

88. Morris, M., Ryall, K., Shen, C., Forlines, C., Vernier, F. (2004) Beyond "Social Protocols": Multi User Coordination Policies for Co-located Groupware. Proceedings of ACM CSCW '04, pp. 266-265.

89. Myers, B., Bhatnagar, R., Nichols, J., Peck, C., Kong, D., Miller, R., and Long, C. (2002) Interacting At a Distance: Measuring the Performance of Laser Pointers and Other Devices. Proc CHI'02, 33-40.

90. Myers, B., Stiel, H., and Gargiulo, R. (1998): Collaborations using multiple PDAs connected to a PC. Proc ACM CSCW'98, ACM Press. 285-294.

91. Nacenta, M. A., Pinelle, D., Stuckel, D., and Gutwin, C. (2007) The effects of interaction technique on coordination in tabletop groupware. Proc. GI '07, ACM Press, 191-198.

92. Nacenta, M. A., Sallam, S., Champoux, B., Subramanian, S., and Gutwin, C. (2006) Perspective cursor: perspective-based interaction for multi-display environments. Proc. CHI '06. ACM Press, 289-298.

93. Oh, A., Fox, H., Kleek, M., Adler, A., Gajos, K., Morency, L., Darrell, T., (2002) Evaluating Look-to-Talk: A Gaze-Aware Interface in a Collaborative Environment. In Extended Abstracts of ACM CHI 02

94. Olsen, D. and Neilsen, T., (2001), Laser Pointer Interaction, Proc. CHI '01, ACM Press, 17-22.

95. Oviatt, S. Multimodal interactive maps: Designing for human performance. *Human-Computer Interaction* 12, 1997.

96. Oviatt, S. L. Ten myths of multimodal interaction, *Comm. ACM*, 42(11), 1999, 74-81.

97. Papineni, K. Roukos, S., Ward, R. (1997) Feature-based language understanding. In European Conf. on Speech Communication and Technology, 1435–1438.

98. Parker, K., Mandryk, R., Nunes, M., Inkpen, K. (2005) TractorBeam Selection Aids: Improving Target Acquisition for Pointing Input on Tabletop Displays. Proc. Interact ’05.

99. Patten, J., Ishii, H., Hines, J., and Pangaro, G. (2001) Sensetable: a wireless object tracking platform for tangible user interfaces. Proc. CHI '01. ACM Press, 253-260.

100. Rekimoto, J., Ullmer, B., and Oba, H. (2001) DataTiles: a modular platform for mixed physical and graphical interactions. Proc. CHI '01, ACM Press, 269-276.

101. Rekimoto, J. (1997) Pick-and-drop: a direct manipulation technique for multiple computer environments. Proc. UIST '97. ACM Press, 31-39.

102. Rekimoto, J. and Saitoh, M. (1999) Augmented surfaces: a spatially continuous work space for hybrid computing environments. Proc. CHI '99. ACM Press, 378-385.

103. Rekimoto, J. (2002) SmartSkin: an infrastructure for freehand manipulation on interactive surfaces. Proc. CHI '02. ACM Press, 113-120.

104. Richardson, T., Stafford-Frasser, Q., Wood, K. and Hopper, A. (1998) Virtual network Computing. IEEE Internet Computing, 2(1), 33-38.

105. Rogers, Y. and Lindley, S. (2004) Collaborating around vertical and horizontal large interactive displays: which way is best? Interacting with Computers (16), 1133-1152. Elsevier.

106. Roussel, N. (2001) Exploring new uses of video with videoSpace. Proc. EHCI'01, vol. 2254, Springer, 73-90.

107. Ryall, K., Forlines, C., Shen, C., Morris, M. (2004) Exploring the Effects of Group Size and Table Size on Interactions with Tabletop Shared-Display Groupware. Proceedings of CSCW 04, pp. 284-293.

108. Scott, S.D., Grant, K.D., and Mandryk, R.L. (2003) System Guidelines for Co-located Collaborative Work on a Tabletop Display. Proc. ECSCW 2003, Springer, 159-178.

109. Scott, S.D., Carpendale, M.S.T, Inkpen, K.M.: Territoriality in Collaborative Tabletop Workspaces. In Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW)'04, 2004, pp. 294–303

110. Segal, L. Effects of checklist interface on non-verbal crew communications, NASA Ames Research Center, Contractor Report 177639. 1994

111. Schmidt, R., Penner, E., Carpendale, M. S. T. (2004) Reconfigurable Displays. Workshop on Ubiquitous Display Environments; at UBICOMP 2004. ACM Press,

112. Shen, C., Vernier, F.D., Forlines, C., Ringel, M. (2004) DiamondSpin: An Extensible Toolkit for Around-the-Table Interaction, Proc. CHI 2004, ACM Press, pp. 167-174.

113. Shoemaker, G. B. D., Inkpen, K. M. (2000) MIDDesktop: An application framework for single display groupware investigations. Technical Report TR 2000-01, School of Computing Science, Simon Fraser University.

114. Sommer, R. (1969) Personal Space The Behavioural Basis of Design, Prentice Hall, Englewood Cliffs, New Jersey, ISBN 0-13-657577-3.

115. Ståhl, O., Wallberg, A., Söderberg, J., Humble, J., Fahlén, L. E., Bullock, A., and Lundberg, J. (2002) Information exploration using The Pond. Proc. CVE '02. ACM Press, 72-79.

116. Stefik, M., Bobrow, D., Foster, G., Lanning, S., Tatar, D. (1987) WYSIWIS revisited: early experiences with multiuser interfaces. ACM TOIS, 5(2), 147–167.

117. Stewart, J., Bederson, B., Druin, A. (1999) Single display groupware: A model for co-present collaboration. ACM CHI'99, 286-293.

118. Strauss, A. and Corbin, J. (1998). Basics of qualitative research, Techniques and Procedures for Developing Grounded Theory, Second Edition, SAGE Publications. Chapter 8: Open Coding, pp 101-122.

119. Streitz, N., Gießler, J., Holmer, T. and Konomi, S. (1999), i-LAND: An interactive Landscape for Creativity and Innovation, Proc. CHI '99, ACM Press, pp. 120-127.

120. Sturman, D.J., Zeltzer, D. (1994) A survey of glove-based input, Computer Graphics and Applications, IEEE, 14(1), 30-39.

121. Subramanian, S., Aliakseyeu, D., and Lucero, A., (2006) Multi-Layer Interaction on Digital Tables, Proc. UIST '06, ACM Press, 269-272.

122. Sutherland, I. E. (1964) Sketch pad a man-machine graphical communication system. Proc. SHARE Design Automation Workshop, DAC '64. ACM Press, 6.329-6.346.

123. Tandler, P., (2003), The BEACH Application Model and Software Framework for Synchronous Collaboration in Ubiquitous Computing Environments, Journal of Systems & Software, Special Edition on Application Models and Programming Tools for Ubiquitous Computing, October, 2003.

124. Tandler, P., Prante, T., Müller-Tomfelde, C., Streitz, N., Steinmetz, R. (2001) Connectables: dynamic coupling of displays for the flexible creation of shared workspaces. Proc. UIST '01, ACM Press, 11-20.

125. Tang, A., Boyle, M. and Greenberg, S. (2005). Display and Presence Disparity in Mixed Presence Groupware. Journal of Research and Practice in Information Technology, Vol. 37, No. 2, May, 71-88.

126. Tang, A., Neustaedter, C. and Greenberg, S. (2006a) VideoArms: Embodiments for Mixed Presence Groupware. Proc. the 20th BCS-HCI British HCI 2006 Group Conference (Sept 11-15, Queen Mary, University of London, UK).

127. Tang, A., Tory, M., Po, B., Neumann, P., and Carpendale, M. S. T. (2006b). Collaborative Coupling over Tabletop Displays. Proc.CHI '06. (April 24-27, Montreal, Quebec). pp: 1181-1190. ACM Press.

128. Tang, J.C. (1991). Findings from observational studies of collaborative work. *International Journal of Man-Machine Studies*, 34, pp. 143-160.

129. Tatar, D., Foster, G. and Bobrow, D. (1991), Design for conversation: lessons from Cognoter, Proc. ECSCW '91, ACM Press, pp. 55-79.

130. Tse, E., Histon, J., Scott, S., Greenberg, S. (2004a) Avoiding Interference: How People Use Spatial Separation and Partitioning in SDG Workspaces. Proceedings of ACM CSCW '04.

131. Tse, E. and Greenberg, S. (2004b) Rapidly Prototyping Single Display Groupware through the SDG Toolkit, Proc. Australasian User Interface Conference, Australian Computer Society Inc., p101-110.

132. Tse, E. (2004c) The Single Display Groupware Toolkit. MSc Thesis, Department of Computer Science, University of Calgary, Calgary, Alberta, Canada, November.

133. Tse, E., Shen, C., Greenberg, S. and Forlines, C. (2006a) Enabling Interaction with Single User Applications through Speech and Gestures on a Multi-User Tabletop. *Proc. AVI 2006.*

134. Tse, E., Greenberg, S., Shen, C. and Forlines, C. (2006b) Multimodal Multiplayer Tabletop Gaming**.** *Proc. Workshop on Pervasive Games* 2006.

135. Tse, E., Greenberg, S. and Shen, C. (2006c) GSI DEMO: Multiuser Gesture / Speech Interaction over Digital Tables by Wrapping Single User Applications. Proc. ICMI'06, ACM Press.

136. Tse, E., Shen, C., Greenberg, S., and Forlines, C. (2007a) How pairs interact over a multimodal digital table. Proc. CHI 07, ACM Press, 215-218.

137. Tse, E., Greenberg, S., Shen, C., Barnwell, J., Shipman, S. and Leigh, D. (2007b) Multimodal Split View Tabletop Interaction Over Existing Applications. Proc. IEEE Tabletop 2007, In Press.

138. Vogel, D., Balakrishnan, R. (2005). Distant freehand pointing and clicking on very large high resolution displays. Proc. ACM UIST 2005, 33-42.

139. Vogel, D. and Baudisch, P. (2007) Shift: A Technique for Operating Pen-Based Interfaces Using Touch. Proc. CHI 2007. p. 657-666.

140. Vogt, F., Wong, J., Fels, S. and Cavens, D. (2003), Tracking Multiple Laser Pointers for Large Screen Interaction, Extended Abstracts of the UIST '03, ACM Press, pp. 95-96.

141. Wellner, P. (1991) The DigitalDesk calculator: Tangible manipulation on a desktop display. Proc. UIST '91, ACM Press, 27-33.

142. Westerman, W., Elias, J. G., and Hedge, A. (2001) Multi-Touch: A New Tactile 2-D Gesture Interface for Human-Computer Interaction. Proc. Human Factors and Ergonomics Society 45th Annual Meeting, 632-636.

143. White, W. (1965) Method for Optical Comparison of Skin Friction-Ridge Patterns. U.S. Patent 3,200,701. Aug. 1965.

144. Wigdor, D., Balakrishnan, R. (2005). Empirical investigation into the effect of orientation on text readability in tabletop displays. Proc. ECSCW '05.

145. Wigdor, D., Shen, C., Forlines, C., Balakrishnan, R., (2006). Table-centric interactive spaces for real-time collaboration: solutions, evaluation, and application scenarios. Proc. CollabTech 2006, 9-15.

146. Wilson, A. D. (2004) TouchLight: an imaging touch screen and display for gesture-based interaction. Proc. ICMI '04. ACM Press, New York, NY, 69-76.

147. Wren, C., Azarbayejani, A., Darrell, T., Pentland, A. (1997) PFinder: Real-Time Tracking of the Human Body, IEEE Transactions on Pattern Analysis and Machine Intelligence ,vol. 19, no. 7, pp. 780-785, July, 1997.

148. Wu, M., Shen, C., Ryall, K., Forlines, C., Balakrishnan, R. Gesture Registration, Relaxation, and Reuse for Multi-Point Direct-Touch Surfaces. IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TableTop), January 2006. pp.183-190 (IEEE Computer Society P2494, ISBN 0-7695-2494-X)

149. Wu, M., Balakrishnan, R. (2003). Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. Proceedings of ACM UIST 2003. pp. 193-202.

150. Yuizono, T., Munemori, J., and Nagasawa, Y. (1998) GUNGEN: Groupware for a New Idea Generation Consistent Support System. Proc. APCHI '98, IEEE Computer Society, 357-363.

Appendix A. Multimodal Pilot Study Codes

Legend

As described in the multi user multimodal pilot study Chapter 3, videos of each pair were recorded and a transcription of each participant's activities was recorded by the experimenter. For the study each speech and gesture command was coded as assistance, validation, or confirmation. These codes are included below for your reference. Each participant is indicated by their seating position. The left (L:) person was sitting on the left side from the view of the camera (or the right side of the digital table) and right (R:) person was seated on the right (or left side of the digital table).

The codes are broken up into two parts and are denoted with capital letters and square braces. For example, CS[CV]. The beginning part represents the modalities used: Speech (S), Gesture (G), and a code if the command was performed across persons (C).

The letters inside the square braces [ ] indicate if the command was used for Confirmation (C), Validation (V), or Assistance (A).

*beep was an audio cue that occurred when the system recognized a speech command.

Group 1 – The Sims

CS[CV] L: computer zoom in *beep
CS[CV] R: computer zoom out (points finger to living room opens up sims character) hmm...
S[C] R: (points to display) Computer second floor *beep computer second floor *beep
CS[CV] L: computer walls down *beep
SG[c] L: computer create... exercise machine *no beep
SG[CV] R: computer create bed places it
CSG[CV] L: computer create night stand *beep bing!
SG[CA] L: lets see... computer create couch *no beep (touches on bar and drags couch into game, other person looking)
G[] L: it's okay, you can drag em. lets add a lounge chair (drags another item)
SG[CV] R: laughs... computer create phone (touch)
SG[CV] L: nice... lets see, washroom we need a washroom... computer create sink *beep okay (places)
G[C] R: alright pan to
CS[CV] R: computer first floor *beep

S[CV] R: computer create fridge *beep whoa (finger pulls back up in shock)
CG[A] L & R: (both move hands towards each other to try to correct this error when within one feet of each other they both SG[CV] L: okay, computer create dishwasher *beep (touch)
SG[C] L: computer create trash can *beep (touch) all the necessities of modern li..
SG[CA] R: computer create microwave (touches twice, but does not place the item)
CS[A] L: computer create counter (autoplaced in their last touch location) now.
S[VA] R: computer create microwave (reaches hand out to touch the display)... *no beep (receds hand) computer create
CSG[CV] SG[C] R: computer create tree (finger) oh, what is that?
CG[A] (picks it up using five fingers almost as if to show to the other person)
S[CA] L: computer delete. *laughs *no beep you go at it
CG[A] R: computer delete *beep
SG[C] L: hmm... computer create.. stove *beep [places in crashes) grrr... (uses five fingers several times to move it into SG[C] R: (watching attentively) hmm.. so.. (L: put it over here) computer create playground *beep oh, so is that what that SG[] L: awesome.. that's pretty sweet piece of.. okay what else do I have? we have a...living room ... we haven't covered
SG[C] R: computer create pool table (L nods) *no beep computer create pool table *beep (single finger placement)
SG[C] L: computer create piano (single finger place) yeah, always a baby grand or something, that sweet so there's a...
SG[CA] Computer create computer *no beep were computer scientists we need computers. create. computer create computer. (L & CSG[A] L: oh it needs to be.. it needs a table. (R: ahh) computer create table *beep (R places the table possibly because he
CSG[CA] R: computer create computer *beep (starts placing the table while L speaks)
S[C] L: **** second floor (R: compu) computer second floor *beep heh we can both say it
SG[C] R: computer create bed *beep (single finger)
SG[C] R: computer create night stand *beep single finger place
SG[] L: computer create table *beep (single finger) computer create computer *beep (reaches arm out, auto places in the
CS[V] R: computer first floor *no beep computer first floor *beep ah right there
G[C] L: you think so? well lets put part of the kitchen in there (both start trying to touch at the same time, system does
G[C] L: nah nah... its all good starts tapping in the kitchen area. All our pieces.. I'll grab.. over there... [moves
SG[C] R: computer create bed *beep (finger)
SG[C] L: computer create diving board *beep oh what am I... we already have a diving board. umn...
SG[C] L: [both look to reference sheet) computer create bbq *beep we always need a barbeque
SG[C] R: computer create television *beep (auto places near location of BBQ raises palm to ceiling single finger moves into
G[C] L: lets move it back in here (moves the bbq)
SG[C] L: showers, there's no showers or bathtub (R: oh yeah) computer create bathtub... yeah we did put it upstairs lets see
S[C] computer second floor *no beep computer second floor *beep (touches to place bathtub) okay there you go
CSG[CV] R: computer create sink *beep (touch) see?
SG[CV] L: computer create shower *no beep computer create shower *beep (touches) bam

G[C] L: hmm (two finger pans) mm hmm...
SG[CA] R: this is great computer create fence. holy nice
CSG[A] L: it went to the.. the just reset it. computer create fence *no beep is that the fence? yeah.
CS[A] L: oh, computer first story err sorry first floor *no beep umn.. computer first floor *beep
SG[C] L: computer create fence *beep (waves hand over the buttons to select a fence) point picket fence (draws with a single
G[C] L: it's pretty sweet though, alright lets try to again (tries to drag a corner when only a straight line is permitted)
G[A] L: it's okay you connect these up (starts to let go and drag again) its pretty fast oh we've got some stuff inside the CG[CA] R: lemme (L: patch it up, tries to touch at the same time as L and it jumps between the two points, laughs and then R
CSG[CV] L: oh it.. (sits back, reads the reference sheet while R continues drawing) sweet computer create tree *no beep
G[A] - L was about to explain that you can't interesect the playground but instead R uses his finger to draw around the
SG[CA] R: computer create hottub *beep what was that?
SG[CA] R: (selects the hot tub with five fingers) computer delete *beep
CSG[CV] L: hmm.. computer create hot tub *no beep computer create hot tub *beep hmm.. (moves finger around to try to find a S[CV] L: computer zoom out.

Group 1 – Google Earth

G[] R: okay (starts zooming the map)
CSG[A] R: computer measure distance *beep (moves window out of way) where is that again.. central parl
G[V] R closes menu
CG[C] L: ghostbusters fire house that's awesome heh, (R: cool) lets see diddy's building aw man, lets see what do you think S[C] L: okay, so lets see... okay, so lets... computer scratch pad *no beep computer scratch pad *beep okay
G[V] R: (draws circle around the statue of liberty) number one
G[C] R: (draw cricle around ground zero and writes a two) square...ll two
CG[A] L: *beep okay (starts panning, tries to draw but starts panning instead)
SG[C] L: or something close to it (continues panning)... computer scratch pad *beep central park..(draws cricle and three, S[V] L: okay *beep
CG[V] R: (starts zoom on the corner of the display closest to him two hands at frst for coarse movement then one hand for
CS[V] R: computer fly to boston *no beep computer fly to boston *beep
G[C] R: so many places to go (two figner zoom)
CG[C] L: yes.. heh.. oh lets go check out boston over here. See there's a bunch of stufff (uses two fingers to zoom) boston G[C] R: oh yeah that's a good place to be (reaches over to the other person's space to point to where the aquarium is)
CG[C] L: the constitution? ah ok. (pans on the display, while the other person does mostly hovering over the display) yeah he
CS[V] R: computer layer buildings *beep
S[A] L: (zooms and pans) lets see, lets check out the prudential center. Computer reset orientation

CS[A] L: computer reorient *beep hmm science park... we definitely want science park
G[C] R: cambridge center... galleria (starts zooming)
CG[C] L: hmm... boston common (hover points to it) is definitely, so this whole area I think is a good place to go so we can
CG[C] R: (steals control from L by panning) yeah okay yeah
S[C] L: so, computer scratch pad *no beep computer scratch pad *beep
G[C] L: I think boston common area (waves hand over boston common) *beep,
CS[A] R: computer scratch pad *beep
CG[C] - both L & R start drawing simultaneously circles in their respective areas of interest
CG[C] L: lets see, boston common's a beautiful place (draws a one by his circle, tries to correct the tip of it,
CG[C] while R draws a two) that's an ugly one (erases with a hand, redraws the one) yup two so go boston common get set up
S[V] R: okay *beep
G[C] L: Umn the last stop because it's on the way, we need the aquarium (starts panning and zooming looking for it)
CG[C] R: It's just.. (takes control and starting moving to help with the finding of the aquarium)
SG[V] L: okay so lets zoom.. lets do a scratch pad there.. computer scratch pad *beep (draws cricle around the supposed
CG[C] -R starts drawing a long line, not sure if it is a fish or just getting bored.
CS[V] L: *sniff *beep okay uh
S[C] L: computer fly to chicago *no beep computer fly to chicago *beep see...
S[C] L: computer fly to millenium park *no beep *laughs (starts panning and zooming) uh what's a cool place?
G[C] L: there's all this stuff down here (starts zooming map)
S[C] R: computer layer shopping
CG[C] - speech was done while the other person was panning the map at the same time
SG[C] L: saks fifth avenue oh my god okay umn... computer scratch pad *no beep computer scratch pad *beep (draws a circle
S[V] L: compute.. okay *no beep okay *beep.
S[C] L: computer layer buildings *beep but he should go to millenium, maybe sears tower, that's a.. there's a building.
S[C] L: coputer layer shopping *beep
G[C] L: I'm all about using this G[C] (tilts the display with five fingers)
S[C] L: computer layer buildings *beep (two finger zoom) okay the pier through this, look for the sears tower
S[G] L: computer scratch pad *beep,
CG[C] R: so... it's far for here.. but there it is. [points to the building]
CSG[A] L: well lemme see lemme measure distance computer measure distance *beep (draws line)
CG[V] R: probably clear the line or something first
S[C] L: computer reorient *beep
G[C] (starts panning)
SG[C] R: computer scratch pad *beep (draws circle around sears tower)
CS[V] L: okay *beep (two finger zoom and pan) maybe he'd like something to eat when he gets here
S[C] L: hehe he wants to eat michael jordan computer... computer layer dining *beep

CG[V] R: (pans and zooms) there's gotta be something around that area
CS[C] R:computer scratch pad *no beep computer scratch pad *no beep computer scratch pad *beep
CS[C] L: computer layer dining
CS[A] R: cancel
CG[C] L: hmm... so we just have to center it whereever we want to go (R pans while L talks) panda garden, okay panda express
CSG[v] R: computer scratch pad *beep (circles restuarant) *beep
SG[C] R: computer scratch pad *beep (L pulls finger back, R moves arm forward to complete drawing and writes a four then
S[V] okay *beep
S[C] R: computer reorient *beep ah, it's already.. why am I doing that?
CS[C] L: computer layer buildings *beep
S[C] L: computer... fly to paris

Group 2 – Google Earth

S[C] L: Computer Fly to New York *beep
S[C] L: Computer Show attractions, computer show... (flinches head) layer attractions
G[] R is zooming the map while this command is being said
CS[V] L Computer layer attractions *beep... ah good, maybe now can go out (starts using two hands to pan)
S[C] L: computer scratch pad *beep so, what do we have here so, I think statue of liberty is almost obligatory heh
G[C] L: so, (starts drawing on table circle with number) ok, so we don't know about the ordering maybe so far (starts erasing
CG[C] R: yeah, just circle both (motions with a circling motion over the table display, L starts circling)
G[A] R puts palm out to start erasing on the table, removes L's recent circle around broadway and starts marking his own
CG[C] L: yeah, broadway goes through (finger pointing to broadway) all this
CG[C] R: ahh (finger pointing to table with palms facing upward) cause.. (L motions with his finger over the similar area to CG[V] L: yeah times square is a good one (reaches hand out, R pulls back his hand and L starts to draw a cicle)
CS[A] L: you know uh like when we have all the places we want him to visit we can probably, so.. computer okay *beep
CG[C] R starts using two fingers to zoom while L is talking.
G[C] R continues to pan to check out the map.
G[C] R: one here (points on table) then... (uses two fingers to point to a distant location (probably central park)
CG[C] L: so maybe, first (touches on the table)
CG[C] R: Maybe one here (points on the table, to the place that he had just pointed, perhaps L didn't understand the last
G[C] L: oh its...(points to the label)
G[V] R: statue of liberty then... (starts panning with a single finger. Both R and L are actively touching on the table at G[A] -for some reason some pop up appears on the screen, L tries to close the pop up.
S[C] L: so, maybe here, united nations, computer scratch pad *no beep

CS[A] R: computer scratch pad *beep there we go.
G[C] R: so, can we do this? (tries to do a two finger zoom when in scratch pad mode) no it doesn't work
CS[A] L: yeah, because you know... (mumbles) scratch pad, uh so... computer okay *no beep computer okay *beep
CG[C] R: well it's (both people start zooming at the same time)
G[C] (while R is zooming):
CS[C] L: computer scratch pad * beep (R pulls hand away before the flash) now okay so we said we start with
CG[V] L: then three (finger circles around) , fourth (writes number), five (number) then that's it and hopefully he will hav
S[V] L: okay, computer okay *beep
S[C] L: computer fly to boston *no beep
CS[A] R: lemme try, computer fly to boston *beep
CG[C] - R starts panning the map while L is ready. L pull the map back for better reading
G[C] R: just let me see it again (starts panning on the map with a single finger,
G[V] L: Yeah, the aquarium is good (touches on the table and takes control of the floor,
CG[C] then R comes in and takes control and the
S[C] L: yeah MIT, is one of the... and MERL of course *both laugh okay so, computer scratch pad *beep
G[V] L: okay, so we decided about (circle) boston tea party yeah we said that (circle) aquarium is nice
CG[C] R: (cicles MIT museum) museum
G[V] L: yeah boston common (cicles) quincy market (circle)
S[V] L: computer okay,
CS[A] R: lemme try.. computer okay *beep
S[A] L: computer cancel *beep computer okay
S[V] L: computer scratch pad *beep and here (hover cicles around area, not sure where it is) it is here
CG[C] (R comes and draws a circle over the same location)
G[C] L comes in and erases both circles and draws onE big one
S[A] L: uh, computer okay, computer okay *beep yup now we will slide to (starts panning away from the prudential markings)
G[C] R: (places finger on prudential markings) hey hey where are you (L looks over to see what's happening) oh okay
G[C] L: yeah, I'm just moving here yeah okay
S[C] R: computer scratch pad *no beep computer scratch pad *beep
G[C] R: okay one (reaches over to L's space and draws a one)
G[C] R: the aquarium.. or quincy (hovers hand over quincy to number it)
G[A] L: no but this corner but, I don't know... then aquarium (circle) then he can have a rest in the park, oh you cannot
s[c] L: computer okay *no beep
cs[v] L: okay, computer layer roads *no beep computer layer roads *beep okay so I think it will be yes,
s[c] R: ah, computer undo layer *beep,
s[v] R: computer fly to chicago *no beep chica - oh computer fly to chicago *beep
s[A] L: computer fly to chicago *beep
G[C] R starts trying to pan the map)

CG[C] L: I will unzoom it.. interesting places around chicago. Maybe if he's into literature, he might be interested in S[V] R: sears first, so, computer scratch pad *no beep computer scartch pads *beep
G[V] L: yeah okay (circles) cafe is fine do you know what's on
S[C] L: no I do not think it will okay, computer okay *no beep computer okay *beep
G[C] (touches on the icon to open up more information, opens briefly then closes, tries to reopen)
G[] L: so lets unzoom it a bit so G[C] (map starts flying away L slams whole hand onto the table,
CG[A] R uses a single finger to try and stop it)
S[] for roads, computer find roads *beep (interpreted as computer fly to rome) oh
S[A] L: oh I see, computer fly chicago *no beep computer fly chicago *beep (interpreted as computer fly to kyoto)
CS[A] R: no, he went to kyoto, japan heh, computer fly to chicago *no beep, computer fly to chicago *no beep
CS[A] L: computer fly to chicago *beep
S[C] L: computer layer roads *no beep, computer layer roads *beep
G[] L: now we can unzoom it a bit (two finger zoom,
CG[V] then R starts to use two fingers to zoom) yeah, there is ernst hemmingways' S[C] computer scratch pad *no beep
S[C] computer scratch pad *no beep computer scratch pad *beep (adjusts microphone) computer scratch pad *beep
CG[V] L: computer ... oh here it is I see. (circles) okay so lets now give it some order.
CG[C] R: here (points)
G[C] L: okay here (numbers on table) then two three four,
CG[C] R: lets see if I can write, oh, I can, five is okay
S[C] L: okay, computer okay *beep

Group 2 – The Sims

S[C] L: oh yeah we have a first floor but maybe we should... computer walls down *no beep computer walls down *beep
CS[A] L: computer zoom in *beep
G[C] L: so no, ok, I think should probably decide which room will be which okay (two people two finger zoom at the same time)
S[C] R: computer create couch *beep
S[V] R: computer create couch *no beep computer create couch *beep
G[C] L: yeah umn.. (points to location where couch should be placed) ok lets move this (R has his finger hovering over the touch, the L starts to
CG[A] R: yeah stop, stop there (R touches the couch and moves it to a wall) here?
S[V] L: computer create arm chair *beep (a counter is created) oh what's this?
S[A] L: computer delete *no beep computer delete *beep
S[C] L: computer create chair *no beep (said very slowly) computer create chair *no beep computer create chair *beep yeah ah yes.
CG[C] (touches on the apropriate chair and then on the bar on the bottom) good
G[C] (uses two fingers to zoom and it moves a bit out of control) rrr... wait!
S[V] L: lets place it here. Computer create television *no beep (very slow)... computer create television *beep oh where is it?

CG[A] R: touches on the table
CG[C] L: (adjusts table) ok and so shall we put some small tables?
CS[V] R: oh, uh computer create table *beep some sort of coffee table like this
CG[A] (tries to select a table from the bottom menu but overlaps L's space) L selects the item from the bottom menu and places it on the table
G[C] R (adds a dark rimmed table from the bottom menu and touches on the original table)
S[C] computer delete *no beep computer delete *beep (tries to move the new table into the old location) ah okay now
G[C] (R starts panning and moves his finger away when L says computer)
SG[C] L: computer create lamp *no beep computer create lamp *beep yup and lets place it here (places)
G[C] R: this one? (selects a lamp that needs to be placed on a table and starts touching in the living room)
G[C] R: (touches more items) okay (but item does not place because the item must be on a table)
G[A] R: oh (tries to find a table surface to place it)
G[A] R: (taps and holds on the table) must place what?
CS[C] L: kitchen here? ok lets make it here so.. computer create fridge *no beep computer create fridge *beep
S[C] L: computer create stove *no beep computer create stove *no beep
CG[V] L: ah there it is (touches on stove) so where do we place it
CG[C] R: touches on the table
CG[C] L: moves the stove to the wall.. what's that (touches plant) lets place it here.
G[C] R: now lets (uses single finger to select and place a microwave but can't because it requires a surface to be on)
S[A] L: microwave but... computer create counter *beep
G[C] L: I would like to create along the wall, like here (moves the counter against the wall)
SG[C] computer create counter *beep around here (places with his finger and then tries to rotate)
G[C] R: can you put this on top of the...(touches a piece of grass on top of the counter
CS[C] L: computer create dishwasher *no beep computer create dishwasher *beep (auto places outside of house)
CG[A] R: *laughs (moves the dishwasher back into the kitchen area) there
G[C] L: place it here (moves the stove to a new location)
SG[C] R: computer create microwave *no beep oh it's here (touches on the bottom menu and adds the microwave to the middle of the kitchen)
S[] L: but we have to (touches on a surface) place it on the table. ah there we can have a table there too computer create table
G[C] *beep (L touches location)
SG[C] computer create chair *beep (L touches location)
CG[C] R (selects some items from the bottom menu and touches on the table) ok lets go up stairs
S[] L: yeah, lets go kitchen is basically... oh trash can, computer create trash can *no beep its important computer create trash can *no beep
SG[A] R: let me try, computer create trash can *beep (auto places)
CG[C] L: oh, (selects trash can from the bottom menu) here we go.
S[C] okay, next one, next thing so kitchen bedroom computer second floor *no beep computer second floor *beep

G[C] R (drags a trash can into that same room and another room)
G[C] L: this is the only bedroom here so (drags a mirror into the room) bedroom and
CG[C] R: night stand? (selects it from the menu and places it on the table) oh, we had to put
S[C] L: erm.. computer delete *beep (removes the piano from the bedroom)
S[C] computer create bed *no beep computer create bed *beep here (hovers around area)
G[C] R: oh, where is it? oh, (moves over to bed button and places it on the other room) I didn't see it
SG[V] R: so, whenever you do. Computer create night stand *no beep (touches on the bottom bar)
CS[V] L: computer create wardrobe *no beep
G[C] R adds a couch to the room
CG[C] L moves the couch
SG[V] L: yeah, it will be, so uh, computer create shower *no beep computer create shower *beep (auto places, in the right room but L decides to try
G[C] (R clicks on a sink in the menu and tries to use that to add a sink to the bathroom)
CG[V] R touches on the sink to indicate that it needs to be moved and L completes the move by touching on the new location
G[C] R: (touches on a sink in the menu and adds it to the bathroom)
G[C] L: yeah (touches on the placed sink and moves it to the corner)
S[C] R: computer walls up *beep
S[C] R: I guess we're done, just roofs up, computer roofs up *no beep (wrong word) computer roof up *no beep ah I said roofs
CS[A] L: computer roof up *beep

Group 3 – Google Earth

S[C] R: okay, computer fly to new york *no beep computer fly to new york *beep
G[C] L: so, which city should we start first? (zooms out with two fingers to US level)
CS[V] L: well maybe, okay lets go for computer fly to new york *beep so what do you think is worth looking at?
CS[C] R: umn lets look at..umn.. computer layer... umn attractions *no beep computer layer attractions *beep hmm
G[C] L pans map while scanning the different sites
CG[A] R (places two fingers in the position of times square and zooms into the location, the both LR start panning)
G[V] L (pulls back to times square area:) times square.
G[C] L continues to pan
CS[V] L: computer scratch pad *beep so
G[C] R (comes in and draws a circle around times square, then tries to draw a circle around central park but it doesn't draw) oh..
CG[A] L (comes in and draws a circle around central park) then goes to central park
CG[C] R (tries to draw a two but doesn't work again) no...
G[V] R (pulls chair in draws two) okay
G[V] L: uh.. yeah that's probably I guess the same night with the times square (pans map over times square)
G[C] R: Yeah, it's just like this huge toy store its so much fun ok (pan the map towards central pk)

CG[C] I guess then going up the (pans towards downtown manhattan empire state building... I don't know
CG[C] L (steals control of the display and starts panning)
S[C] well I woul dprobably advice on going here computer scratch pad *beep and then
G[C] (draws a circle around port) flying on a helicopter around downtown new york do you think?
CS[V] L: computer okay *beep
G[C] (pans map searching for next place) and then
G[C] L: yeah maybe going to the wall street and umn chinatown (pan down to wall street)
CS[V] R: okay comptuer scratch pad *no beep computer scratch pad *beep
G[C] (circle around chinatown number, circle around wall st number drawing some words maybe wall st. then erases with a hand draws a cricle
CS[C] L:computer okay *beep
G[C] L: yeah I guess if they have time they can walk, just wander around so, zooms out
G[C] R: look at that when you zoom out it only shows the important ones or the ones that it thinks is important at least.
G[C] R: yeah its not that good, oh harvard sq, Look at that fenway park
S[C] L: ok computer fly to new york *beep computer fly to chicago *no beep
CS[A] R: (in an insulted tone) comptuer fly to chicago *no beep computer fly to chicago *beep okay
G[C] L: I guess Chicago should be more of an arts visit (pans to centre of chicago downtown area)
G[C] R: oh where is this? picasso sculpture. I don't remember this. Where is that? (two finger zoom and pan right into the sculputure)
CG[C] L: (uses two fingers to zoom out when R steals control)
S[C] R: computer overlay roads *no beep
CS[C] L: computer layer roads *beep
G[A] -LR try to pan but something has broken cause both try to touch at the same time. Flies off L managed to recover by holding on the mouse
CS[A] L: computer fly to chicago *beep
S[C] R: computer undo layer *beep oh I love this hard rock cafe, have you been there?
SG[V] L: computer... scratch pad *beep (draw something then erases it, draws a circle around cafe) hard rock cafe
G[C] R starts erasing the numbers and renumbers them) so
G[C] R: if we zoom in (two finger zoom) will it still be?
CG[A] L (completes the erased circle of R)
CS[V] R: computer okay *beep now if we zoom in will it still be around navy pier
G[C] (uses two finger to try to zoom, both LR trying to zoom/pan at same time)
CG[A] L (zooms for R since it doesn't seem to be working for her) yes
G[C] R (uses a whole hand to pan and then tries to zoom but fails)
G[A] L (zooms out for R)
G[C] L: its right over here (two finger zoom) you see music milleniu... (zooms more, pans and fine tues location for the park)
SG[V] L: computer scratch pad *beep (draws a circle around the park) this is roughly
CSG[C] R (draws some initials) okay, computer okay *no beep computer okay *beep
G[C] L: ok lets leave that for Boston (pans map around to view items)
G[C] R (pans map around chicago looking for MOMA) I can't see the museum of art, I guess going to *** street is fine

S[C] L: computer layer *** *no beep computer undo layer *no beep computer undo layer *beep
S[V] L: (validates what is spoken by experimenter) computer layer attractions *beep oh okay, it works,
S[C] L: computer layer buildings *no beep computer layer buildings *beep
G[C] (two finger zoom in, pan to millenium park area, uses five fingers to tilt) so,
G[C] L (tilts with five fingers)
G[C] L (continues to use five fingers to pan) I would actually say that it's right over here
G[A] L: we'll it's close (uses five fingers to bring back a top down view), it is on umn..
G[C] L: this is probably it (touches on table) this is the cotinuation (circles on table) and this is the chicago arts museum
CSG[V] R: okay, computer scratch pad *no beep computer scartch pad *beep (circles the area and annotates with text)
CS[V] L: computer okay *beep (zooms out)
G[C] L: (continues to zoom out) computer fly to boston *beep
G[V] L: umn... (zooms out to see harvard area, then zooms back in)
S[C] L: computer layer terrain *beep
S[C] L: mmm... computer layer buildings *beep okay computer layer terrain *beep, so
S[C] R: yeah, computer layer community bookmarks *no beep computer layer community bookmarks *no beep what is community bookmarks?
S[A] L: computer layer community bookmarks *beep (acidentally added the roads layer instead) hmm....
S[C] L: computer layer roads *beep yeah that was roads *laughs
G[C] R: okay, so fine, lets find harvard. It's right here right (starts panning map) toward harvard
SG[C] L: computer layer buildings (five finger tilt)
S[C] R: attractions, computer layer attractions *no beep computer layer attractions *beep
G[C] (uses a whole hand to tilt the map back upwards, two finger zoom, then one finger pan)
G[C] R: where are we? yeah that's what I'm thinking. Okay we know that (points to a distant location while panning with map)
CG[C] L: ok, massachusetts (zooms map out) ok harvard sq. ok this,
S[C] R: yeah, this is mass ave. This guy (hover point) computer layer roads *no beep computer layer roads *beep
S[C] R (zooms in and pans the map), so, comptuer layer... no computer scratch pad *no beep say it it likes your voice
CS[A] L: computer scratch pad *beep
CG[V] R (draws a big circle around the harvard campus and then a one)
S[V] L: computer okay *beep
S[C] computer fly to boston *beep oh
S[C] R: computer layer building *no beep computer layer buildings *beep thank you
G[C] R (pans map) there's the baseball, where is fenway park (zooms out, pans and finds it)? okay, how do you?
S[A] L: computer scratch pad *beep
CG[C] R (draws a circle around fenway park) this is number one? number two? (draws number)
S[V] L: computer okay *beep so
G[C] L: yeah I mean if whatever and the whales are here,that should be here (zooms in and pans)

SG[C] L: computer scratch pad *beep so it should be right over here (circle over pier)
S[V] R: computer okay *beep ok so what now?
G[V] L: yeah, I think it's a good thing. He'll see all the historical buildings now, zooms out
S[C] R: computer scratch pad *beep (circles tea party, annotates)
CS[V] L: computer okay *beep ok and some famous clam chowder place
S[C] L: (pans display around) computer dining *no beep computer layer dining *no beep computer layer dining *beep ooh
S[A] R: oh my god, computer layer attractions *no beep computer layer attractions *beep okay
S[C] L: computer layer roads *beep hmmm...
G[C] R: maybe lets look close to the (pans map towards pier) close to the ocean front right (zooms in, pans to find previously annotated area).
G[C] L: umn..(panning around) I don't think you will find anything that's much better there's crowds of shapes anyway
CG[C] R: so (steals panning) I think there's something around here... the aqarium?
G[C] L: oh, the aquarium is here (pans, error occurs and L has to pan then use a bunch of tricks to make it stop) (points to a location)
G[C] R (zooms in to that location)
G[C] L (zooms out to see the restuarants) yeah, it's not the aqarium so, it should be something over here.
G[C] R: lets just pick something (pans map)
S[C] L: computer scratch pad *beep
CG[C] R (draws a circle around the restaurant and some annotation)
S[V] L: computer okay *beep

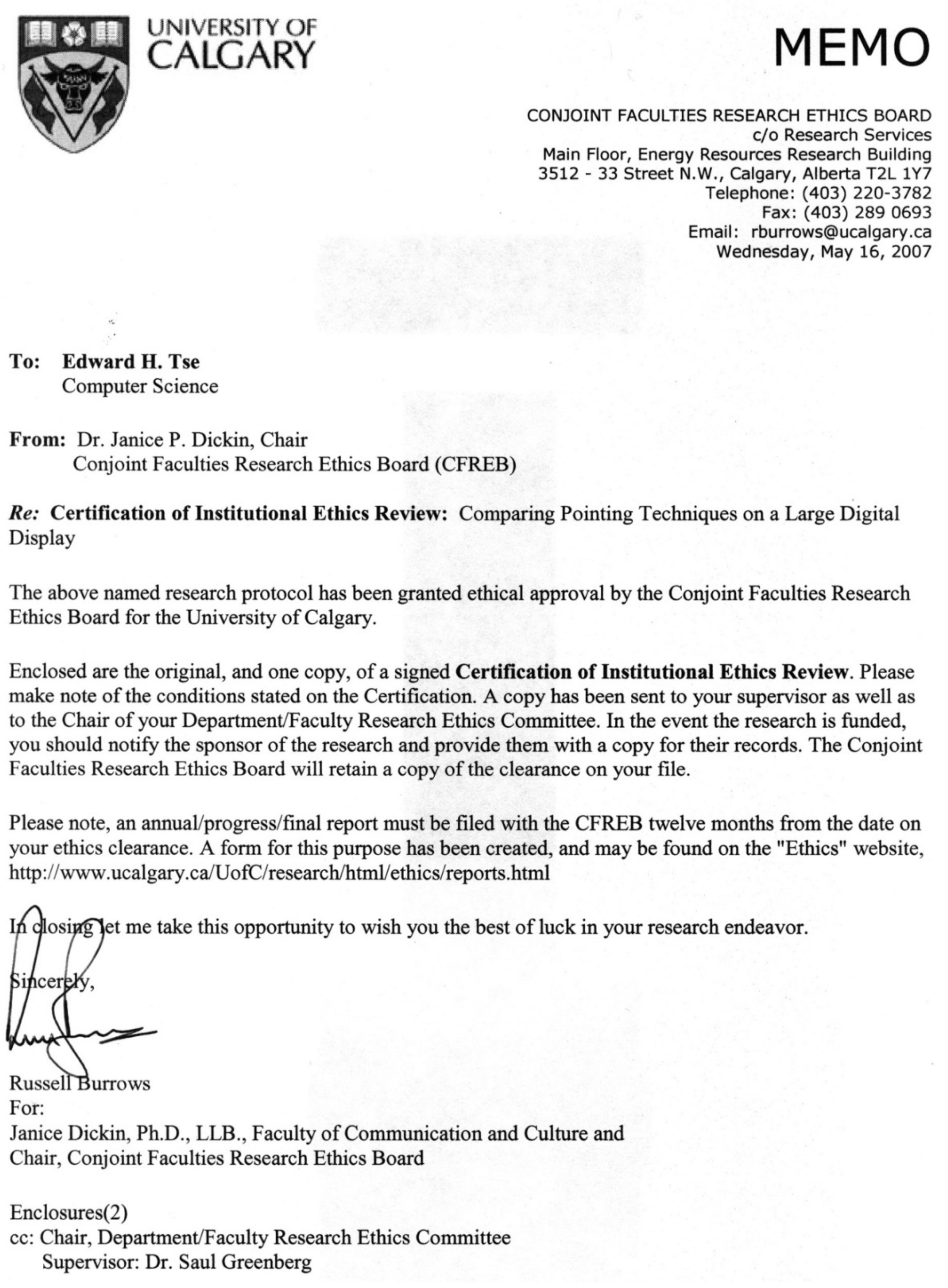
Group 3 – The Sims

G[C] L (uses two finger to pan around the house to get a better view) okay, so lets put it here (hover points to room close to him)
S[V] L: okay uh lets go with it must... home theather system...computer home theater system *no beep computer create home theather system
S[C] L: computer home theater system *no beep computer create home theater system *beep
CG[V] R (touches in the room) what? (autoplaced in the wrong location, R moves it to the new location)
G[A] R (moves the home theater system back to the proposed living room) yeah
S[C] L: computer walls down *beep
S[C] R: okay, computer create uh couch *no beep
S[A] L: computer create couch *beep okay, I would say a couple book shelfs,
S[C] R: are these two couches? (hover points to shrubs) computer create couch *no beep computer create couch *beep
G[C] L: yeah...good (moves the couch to the side of the room) and rotates it
G[C] R: wait.. no I want it here, and turn around. (moves couch and rotates)
s[c] L: computer create book shelf *no beep computer create book shelf *beep (points)
s[c] R: computer show doors? *no beep can you say show? computer doors? computer doors? how do you?
S[A] L: computer walls up *beep

G[V] L: okay, I see (moves the bookshelf into a different location, picks up the fireplace accidentally)
G[A] L: (oh, touches into new location, pans out of control and then pans back into right location, touches wrong objects)
S[C] R: okay, lets do the bathroom and the washroom, computer create sink *beep (tap, but doesn't place)
G[A] L (taps to correctly place)
SG[C] computer walls down *beep computer zoom in *beep computer zoom in *beep (pans to washroom) okay
SG[C] R: okay, computer create uh tub *no beep computer create bathtub *no beep computer create bathtub *beep (places, moves to new location)
S[C] R: computer okay *beep (deletes the bathtub) computer create bathtub *beep
S[C] R: okay computer create uh shower *beep (auto places, then moves to a new location, but someone touches in the bar and delets it)
CS[A] L: computer create bathtub *beep
G[A] L (places bathtub)
SG[C] computer create counter *beep (places) okay
S[C] L: computer zoom out *beep
S[C] computer second floor *beep two finger pan.
S[C] Lets make this thing a bedroom (hover points to L bed) computer zoom in *beep so what do we want?
S[C] R: uh this is the daughter's bedroom, computer create a bed *beep computer create bed *beep
S[A] L: computer create bed *beep
G[C] R (places but still yellow) oh it's not selected.... (places)
S[C] R: computer create mirror *no beep computer create mirror *beep (single finger place)
S[C] R: comptuer create another mirror *no beep
S[A] L: computer create mirror *beep
CG[V] R (places) okay
SG[C] L: computer create shelf *beep (single finger place)
SG[C] L: computer uh first floor *no beep computer first floor *beep (pans map to kitchen area)
G[C] R (starts panning) around the house
G[C] L: (pans back to kitchen) I think this place is the ktichen area
S[C] L: no, it's indoor computer create fridge *no beep computer create fridge *beep
CG[V] R (places, autoplace in midlde of screen)
SG[C] L computer create stove *beep (places, auto place near R, L uses five fingers to pick up the object and moves it tot he left)
SG[C] L computer create a trash can *beep (touches) probably
SG[C] R: computer create counter *no beep computer create counter *beep (places)
SG[C] R: computer create microwave *no beep computer create microwave *no beep (touches counter, moves counter, and places back) computer create microwave *no beep
S[A] L: computer create microwave *beep
CG[V] R (touches around counter until it actually touches) it doesn't like me does it?
S[C] L: yeah it doesn't umn.. computer create dishwasher *beep (both reach in to touch and realizing that both are about to crash they pull
CG[V] R (reaches to the auto placed location for the dishwasher and moves it close to the counter)

G[C] L: okay, lets move it (moves stove to right besides the dishwasher)
CG[C] R (moves the fridge right besides the counter, so now all the kitchen items are aligned)
S[C] L: umn.. computer zoom out *beep
G[C] R: it just needs that small space (hover two finger hand sides over kitchen area) okay we're done.

Appendix B. Speech Bubble Study Ethics



UNIVERSITY OF CALGARY

MEMO

CONJOINT FACULTIES RESEARCH ETHICS BOARD
c/o Research Services
Main Floor, Energy Resources Research Building
3512 - 33 Street N.W., Calgary, Alberta T2L 1Y7
Telephone: (403) 220-3782
Fax: (403) 289 0693
Email: rburrows@ucalgary.ca
Wednesday, May 16, 2007

To: Edward H. Tse
Computer Science

From: Dr. Janice P. Dickin, Chair
Conjoint Faculties Research Ethics Board (CFREB)

Re: **Certification of Institutional Ethics Review:** Comparing Pointing Techniques on a Large Digital Display

The above named research protocol has been granted ethical approval by the Conjoint Faculties Research Ethics Board for the University of Calgary.

Enclosed are the original, and one copy, of a signed **Certification of Institutional Ethics Review**. Please make note of the conditions stated on the Certification. A copy has been sent to your supervisor as well as to the Chair of your Department/Faculty Research Ethics Committee. In the event the research is funded, you should notify the sponsor of the research and provide them with a copy for their records. The Conjoint Faculties Research Ethics Board will retain a copy of the clearance on your file.

Please note, an annual/progress/final report must be filed with the CFREB twelve months from the date on your ethics clearance. A form for this purpose has been created, and may be found on the "Ethics" website, http://www.ucalgary.ca/UofC/research/html/ethics/reports.html

In closing let me take this opportunity to wish you the best of luck in your research endeavor.

Sincerely,

[illegible]

Russell Burrows
For:
Janice Dickin, Ph.D., LLB., Faculty of Communication and Culture and
Chair, Conjoint Faculties Research Ethics Board

Enclosures(2)
cc: Chair, Department/Faculty Research Ethics Committee
Supervisor: Dr. Saul Greenberg

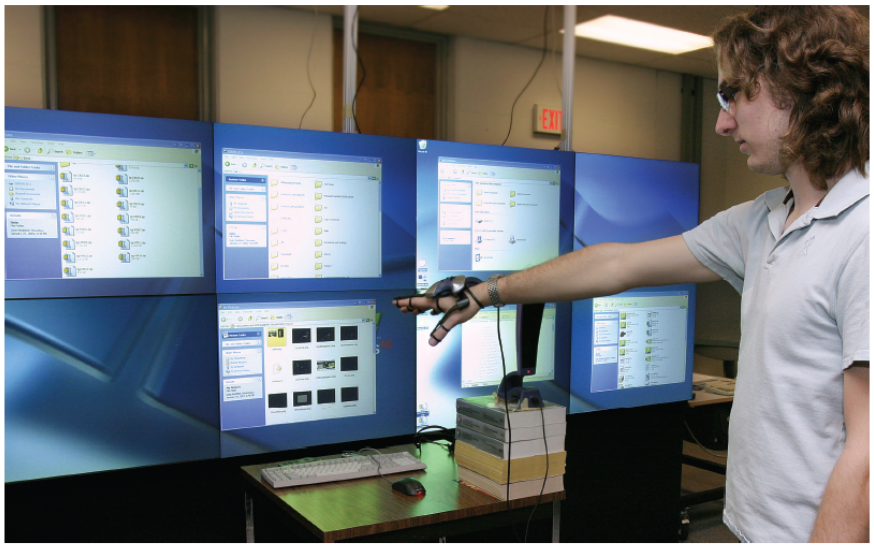
Appendix C. Speech Bubble Study Forms and Questionnaires

Study Recruitment Notice

UNIVERSITY OF CALGARY

Interaction with a Wall Sized Digital Display

Study Recruitment



Investigators: Edward Tse, Mark Hancock, Saul Greenberg

Experiment Purpose: The purpose of this research is to compare the performance of several pointing techniques for large displays. We are looking for women who are not colour blind (with dominant right hand preferred) to participate in a study involving speech and gestures over a large digital wall sized display.

Procedure: You will be asked to perform a number of pointing tasks over a large wall sized digital display.

Objective: The research objective is to compare several selection techniques for large displays to inform the design of future interaction techniques over large digital displays.

Commitment: Your participation in the study will take between 30 and 45 minutes and you will be compensated for your time with a payment of $20.

To Participate or For More Information: Send email to: tsee@cpsc.ucalgary.ca with a date (between April 12-25) and time (9:00am-5:00pm) that you would be able to participate.

UNIVERSITY OF CALGARY

Department of Computer Science
University of Calgary
2500 University Drive
Calgary, AB, CANADA T2N 1N4

CONSENT FORM

Research Project Title: Comparing Distant Freehand Pointing Techniques for Large Displays

Investigators: Edward Tse, Mark Hancock, Saul Greenberg

This consent form, a copy of which has been given to you, is only part of the process of informed consent. It should give you the basic idea of what the research is about and what your participation will involve. If you would like more detail about something mentioned here, or information not included here, you should feel free to ask. Please take the time to read this carefully and to understand any accompanying information.

Purpose: The purpose of this research is to compare the performance of several pointing techniques for large digital displays.

Participant Recruitment and Selection:
To be a recruited for this study, we ask that you be able bodied and already experienced with using a mouse, and that you allow us to use and analyze your results from the study.

Procedure:
The study should require up to one hour of your time. You will be asked to fill out a pre test questionnaire. Then you will be introduced to three distant freehand pointing techniques for a large display. After you are comfortable with each technique you will be asked to perform a series of selections over a wall sized digital display while being video recorded. When you have completed the experiment you will be debriefed by the experimenter and asked to fill out a post test questionnaire.

Confidentiality:
Your anonymity will be strictly maintained. Reports and presentations will refer only to a participant identification number and will be in a secure filing cabinet or on a secure computer. Confidential information will not be used from videos in the publication of results from this study, unless prior consent is given at the end of this consent form.

Risks:
There are no known risks, however, if you feel uncomfortable you are free to quit at any time. All information collected from a person that withdraws during the study session will be destroyed.

Investigators:
Edward Tse and Mark Hancock are PhD students in the Department of Computer Science at the University of Calgary. Dr. Saul Greenberg, is a Professor in the Department of Computer Science.

Your signature on this form indicates that you have understood to your satisfaction the information regarding participation in the research project and agree to participate as a subject. In no way does this waive your legal rights nor release the investigators, sponsors, or involved institutions from their legal and professional responsibilities. You are free to withdraw from the study at any time. Your continued participation should be as informed as your initial consent, so you should feel free to ask for clarification or new information throughout your participation. If you have further questions concerning matters related to this research, please contact:

Edward Tse (tsee@cpsc.ucalgary.ca) or Dr. Saul Greenberg (saul@cpsc.ucalgary.ca)

UNIVERSITY OF CALGARY

Edward Tse
Department of Computer Science
University of Calgary
2500 University Drive
Calgary, AB, CANADA T2N 1N4

If you have any questions or issues concerning this project that are not related to the specifics of the research, you may also contact the Research Services Office at 220-3782 and ask for Mrs. Patricia Evans.

I agree to have videos of my session used in future publications/presentations (Please Initial) _______________

Participant's Name	Date
Participant's Signature	Date
Investigator's/Witness's Signature	Date

A copy of this consent form has been given to you to keep for your records and reference.

Pre-Study Questionnaire

Gender: Male ________ Female ________

Age: ____________________

Are you a Computer Science student / staff / faculty? Yes _______ No _______

Which is your dominant hand (left / right / both)? ______________________

C. How often do you use a computer mouse?

1	2	3	4	5
Rarely	Yearly	Monthly	Weekly	Daily

G. How often do you play video/arcade games?

1	2	3	4	5
Rarely	Yearly	Monthly	Weekly	Daily

F. What is your fluency level with the English language?

1	2	3	4	5
Not fluent	1 year experience	2 years experience	3 years experience	> 3 years

What language do you speak at home?

E. Do you have any physical or visual ailments that might make pointing with a mouse difficult for you (blurred vision, colour blindness)? If so, please describe.

D. Have you ever used large display pointing device (e.g., Nintendo Wii, Smart Board, etc.) If so, please specify which ones.

Post-study Questionnaire

A. I found the ray casting pointing technique easy to use

1	2	3	4	5
Strongly disagree	Disagree	Neutral	Agree	Strongly agree

B. I found the bubble ray casting technique easy to use

1	2	3	4	5
Strongly disagree	Disagree	Neutral	Agree	Strongly agree

C. I found the speech bubble ray casting technique easy to use

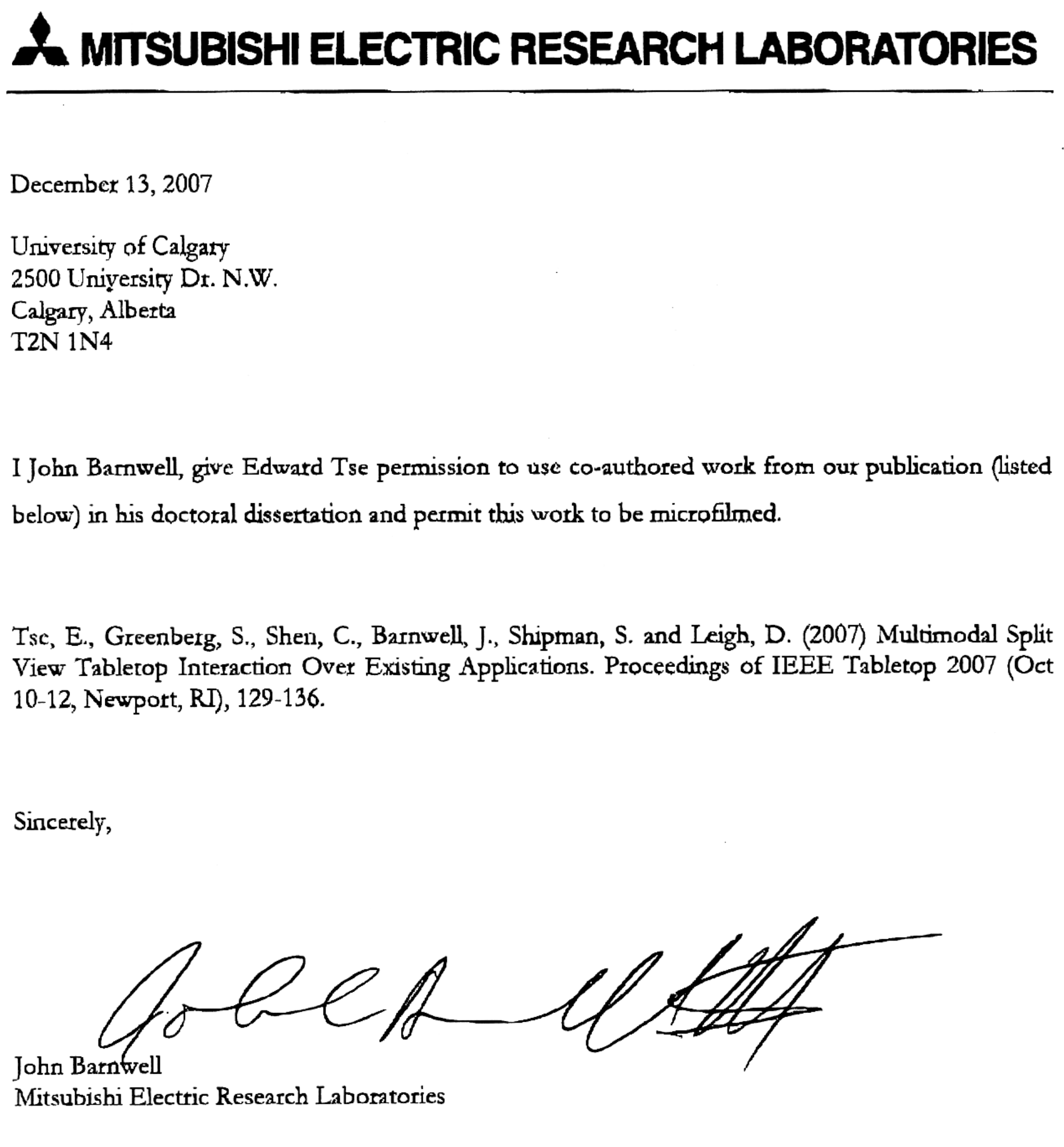
1	2	3	4	5
Strongly disagree	Disagree	Neutral	Agree	Strongly agree

D. Which pointing technique did you most prefer and why?

E. Which pointing technique did you most dislike and why?

F. Do you have any other comments?

Appendix C. Co-Author Permission



MITSUBISHI ELECTRIC RESEARCH LABORATORIES

December 13, 2007

University of Calgary
2500 University Dr. N.W.
Calgary, Alberta
T2N 1N4

I John Barnwell, give Edward Tse permission to use co-authored work from our publication (listed below) in his doctoral dissertation and permit this work to be microfilmed.

Tse, E., Greenberg, S., Shen, C., Barnwell, J., Shipman, S. and Leigh, D. (2007) Multimodal Split View Tabletop Interaction Over Existing Applications. Proceedings of IEEE Tabletop 2007 (Oct 10-12, Newport, RI), 129-136.

Sincerely,

[illegible]

John Barnwell
Mitsubishi Electric Research Laboratories

MITSUBISHI ELECTRIC RESEARCH LABORATORIES

December 13, 2007

University of Calgary
2500 University Dr. N.W.
Calgary, Alberta
T2N 1N4

I Clifton Forlines, give Edward Tse permission to use co-authored work from our publications (listed below) in his doctoral dissertation and permit this work to be microfilmed.

Tse, E., Greenberg, S., Shen, C. and Forlines, C. (2007) Multimodal Multiplayer Tabletop Gaming. In ACM CIE Computers in Entertainment. June. ACM Press (this is a reprint rewarded to the best papers at Pervasive Games)

Tse, E., Shen, C., Greenberg, S. and Forlines, C. (2006) Enabling Interaction with Single User Applications through Speech and Gestures on a Multi-User Tabletop. Proceedings of Advanced Visual Interfaces (AVI'06), May 23-26, 336-343, Venezia, Italy, ACM Press.

Tse, E., Greenberg, S., Shen, C. and Forlines, C. (2006) Multimodal Multiplayer Tabletop Gaming. Proceedings Third International Workshop on Pervasive Gaming Applications (PerGames'06), in conjunction with 4th Intl. Conference on Pervasive Computing, (May 7th Dublin, Ireland), 139-148.

Tse, E., Shen, C., Greenberg, S. and Forlines, C. (2007) How Pairs Interact Over a Multimodal Digital Table. Proceedings of ACM CHI Conference on Human Factors in Computing Systems (May 1-3, San Jose, CA), ACM Press, 215-218.

Tse, E., Greenberg, S., Shen, C., Forlines, C., and Kodama, R. (2007) Exploring True Multi-User Multimodal Interaction over a Digital Table, Proceedings of ACM DIS Conference (Feb 25-27, Cape Town, South Africa).

Sincerely,

[illegible]

Clifton Forlines
Mitsubishi Electric Research Laboratories

UNIVERSITY OF CALGARY

Saul Greenberg
NSERC / iCORE / SMART - Chair In Interactive Technologies
Department of Computer Science, University of Calgary
Calgary, Alberta, CANADA T2N 1N4
phone: +1 403 220 6087 fax: +1 403 284 4707
email: saul.greenberg @ ucalgary.ca

December 13, 2007
University of Calgary
2500 University Dr. N.W.
Calgary, Alberta
T2N 1N4

I Saul Greenberg, give Edward Tse permission to use co-authored work from our publications (listed below) for use in his doctoral dissertation and permit this work to be microfilmed.

Journal Articles

Tse, E., Greenberg, S., Shen, C. and Forlines, C. (2007) Multimodal Multiplayer Tabletop Gaming. In ACM CIE Computers in Entertainment. June. ACM Press (this is a reprint rewarded to the best papers at Pervasive Games)

Papers

Tse, E., Shen, C., Greenberg, S. and Forlines, C. (2006) Enabling Interaction with Single User Applications through Speech and Gestures on a Multi-User Tabletop. Proceedings of Advanced Visual Interfaces (AVI'06), May 23-26, 336-343, Venezia, Italy, ACM Press.

Tse, E., Greenberg, S., Shen, C. and Forlines, C. (2006) Multimodal Multiplayer Tabletop Gaming. Proceedings Third International Workshop on Pervasive Gaming Applications (PerGames'06), in conjunction with 4th Intl. Conference on Pervasive Computing, (May 7th Dublin, Ireland), 139-148.

Tse, E., Greenberg, S. and Shen, C. (2006) GSI DEMO: Multi User Gesture / Speech Interaction over Digital Tables by Wrapping Single User Applications. Proc Eighth International Conference on Multimodal Interfaces (ICMI'06), (Nov 2-4, Banff, Canada), ACM Press, 76-83.

Tse, E., Shen, C., Greenberg, S. and Forlines, C. (2007) How Pairs Interact Over a Multimodal Digital Table. Proceedings of ACM CHI Conference on Human Factors in Computing Systems (May 1-3, San Jose, CA), ACM Press, 215-218.

Tse, E., Hancock, M. and Greenberg, S. (2007) Speech-Filtered Bubble Ray: Improving Target Acquisition on Display Walls. Proceedings of ICMI 2007 (Nov 12-15, Nagoya, Japan), 307-314.

Tse, E., Greenberg, S., Shen, C., Barnwell, J., Shipman, S. and Leigh, D. (2007) Multimodal Split View Tabletop Interaction Over Existing Applications. Proceedings of IEEE Tabletop 2007 (Oct 10-12, Newport, RI), 129-136.

Tse, E., Greenberg, S., Shen, C., Forlines, C., and Kodama, R. (2007) Exploring True Multi-User Multimodal Interaction over a Digital Table, Proceedings of ACM DIS Conference (Feb 25-27, Cape Town, South Africa).

Demos, Videos and Posters

Tse, E., Greenberg, S. and Shen, C. (2006) Motivating Multimodal Interaction Around Digital Tabletops. Video Proceedings of ACM CSCW'06 Conference on Computer Supported Cooperative Work, November, ACM Press.

Tse, E., Greenberg, S., Shen, C. (2006) Multi User Multimodal Tabletop Interaction over Existing Single User Applications. Demo, Adjunct Proc ACM CSCW 2006.

Tse, E., Greenberg, S., Shen, C. (2006) Exploring Interaction with Multi User Speech and Whole Handed Gestures on a Digital Table. Demonstration, Adjunct Proc ACM UIST 2006.

Sincerely,

[illegible]

Dr. Saul Greenberg
Professsor
Department of Computer Science
University of Calgary

UNIVERSITY OF CALGARY

December 13, 2007

University of Calgary
2500 University Dr. N.W.
Calgary, Alberta
T2N 1N4

I Mark Hancock, give Edward Tse permission to use co-authored work from our publication (listed below) in his doctoral dissertation and permit this work to be microfilmed.

Tse, E., Hancock, M. and Greenberg, S. (2007) Speech-Filtered Bubble Ray: Improving Target Acquisition on Display Walls. Proceedings of ICMI 2007 (Nov 12-15, Nagoya, Japan), 307-314.

Sincerely,

[illegible]

Mark Hancock
Department of Computer Science
University of Calgary

MITSUBISHI ELECTRIC RESEARCH LABORATORIES

December 13, 2007

University of Calgary
2500 University Dr. N.W.
Calgary, Alberta
T2N 1N4

I Ryo Kodama, give Edward Tse permission to use co-authored work from our publication (listed below) in his doctoral dissertation and permit this work to be microfilmed.

Tse, E., Greenberg, S., Shen, C., Forlines, C., and Kodama, R. (2007) Exploring True Multi-User Multimodal Interaction over a Digital Table, Proceedings of ACM DIS Conference (Feb 25-27, Cape Town, South Africa).

Sincerely,

小玉 亮

Ryo Kodama
Mitsubishi Electric Research Laboratories

MITSUBISHI ELECTRIC RESEARCH LABORATORIES

December 13, 2007

University of Calgary
2500 University Dr. N.W.
Calgary, Alberta
T2N 1N4

I Darren Leigh, give Edward Tse permission to use co-authored work from our publication (listed below) in his doctoral dissertation and permit this work to be microfilmed.

Tse, E., Greenberg, S., Shen, C., Barnwell, J., Shipman, S. and Leigh, D. (2007) Multimodal Split View Tabletop Interaction Over Existing Applications. Proceedings of IEEE Tabletop 2007 (Oct 10-12, Newport, RI), 129-136.

Sincerely,

[illegible]

Darren Leigh
Mitsubishi Electric Research Laboratories

MITSUBISHI ELECTRIC RESEARCH LABORATORIES

December 13, 2007

University of Calgary
2500 University Dr. N.W.
Calgary, Alberta
T2N 1N4

I, Chia Shen, give Edward Tse permission to use co-authored work from our publications (listed below) in his doctoral dissertation and permit this work to be microfilmed.

Journal Articles

Tse, E., Greenberg, S., Shen, C. and Forlines, C. (2007) Multimodal Multiplayer Tabletop Gaming. In ACM CIE Computers in Entertainment. June, ACM Press (this is a reprint rewarded to the best papers at Pervasive Games)

Papers

Tse, E., Shen, C., Greenberg, S. and Forlines, C. (2006) Enabling Interaction with Single User Applications through Speech and Gestures on a Multi-User Tabletop. Proceedings of Advanced Visual Interfaces (AVI'06), May 23-26, 336-343, Venezia, Italy, ACM Press.

Tse, E., Greenberg, S., Shen, C. and Forlines, C. (2006) Multimodal Multiplayer Tabletop Gaming. Proceedings Third International Workshop on Pervasive Gaming Applications (PerGames'06), in conjunction with 4th Intl. Conference on Pervasive Computing, (May 7th Dublin, Ireland), 139-148.

Tse, E., Greenberg, S. and Shen, C. (2006) GSI DEMO: Multi User Gesture / Speech Interaction over Digital Tables by Wrapping Single User Applications. Proc Eighth International Conference on Multimodal Interfaces (ICMI'06), (Nov 2-4, Banff, Canada), ACM Press, 76-83.

Tse, E., Shen, C., Greenberg, S. and Forlines, C. (2007) How Pairs Interact Over a Multimodal Digital Table. Proceedings of ACM CHI Conference on Human Factors in Computing Systems (May 1-3, San Jose, CA), ACM Press, 215-218.

Tse, E., Greenberg, S., Shen, C., Barnwell, J., Shipman, S. and Leigh, D. (2007) Multimodal Split View Tabletop Interaction Over Existing Applications. Proceedings of IEEE Tabletop 2007 (Oct 10-12, Newport, RI), 129-136.

Tse, E., Greenberg, S., Shen, C., Forlines, C., and Kodama, R. (2007) Exploring True Multi-User Multimodal Interaction over a Digital Table, Proceedings of ACM DIS Conference (Feb 25-27, Cape Town, South Africa).

Demos, Videos and Posters

Tse, E., Greenberg, S. and Shen, C. (2006) Motivating Multimodal Interaction Around Digital Tabletops. Video Proceedings of ACM CSCW'06 Conference on Computer Supported Cooperative Work, November, ACM Press.

Tse, E., Greenberg, S., Shen, C. (2006) Multi User Multimodal Tabletop Interaction over Existing Single User Applications. Demo, Adjunct Proc ACM CSCW 2006.

Tse, E., Greenberg, S., Shen, C. (2006) Exploring Interaction with Multi User Speech and Whole Handed Gestures on a Digital Table. Demonstration, Adjunct Proc ACM UIST 2006.

Sincerely,

[illegible]

Dr. Chia Shen
Mitsubishi Electric Research Laboratories

MITSUBISHI ELECTRIC RESEARCH LABORATORIES

December 13, 2007

University of Calgary
2500 University Dr. N.W.
Calgary, Alberta
T2N 1N4

I, Sam Shipman, give Edward Tse permission to use co-authored work from our publication (listed below) in his doctoral dissertation and permit this work to be microfilmed.

Tse, E., Greenberg, S., Shen, C., Barnwell, J., Shipman, S. and Leigh, D. (2007) Multimodal Split View Tabletop Interaction Over Existing Applications. Proceedings of IEEE Tabletop 2007 (Oct 10-12, Newport, RI), 129-136.

Sincerely,

Sam Shipman

Sam Shipman
Mitsubishi Electric Research Laboratories