Technical Report

UCAM-CL-TR-734
ISSN 1476-2986

Number 734

UNIVERSITY OF CAMBRIDGE

Computer Laboratory

Tabletop interfaces for remote collaboration

Philip Tuddenham

December 2008

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

This technical report is based on a dissertation submitted June 2008 by the author for the degree of Doctor of Philosophy to the University of Cambridge, Gonville and Caius College.

ISSN 1476-2986

Tabletop Interfaces for Remote Collaboration

Philip Tuddenham

Effective support for synchronous remote collaboration has long proved a desirable yet elusive goal for computer technology. Although video views showing the remote participants have recently improved, technologies providing a shared visual workspace of the task still lack support for the visual cues and work practices of co-located collaboration.

Researchers have recently demonstrated shared workspaces for remote collaboration using large horizontal interactive surfaces. These *remote tabletop interfaces* may afford the beneficial work practices associated with co-located collaboration around tables. However, there has been little investigation of remote tabletop interfaces beyond limited demonstrations. There is currently little theoretical basis for their design, and little empirical characterisation of their support for collaboration. The construction of remote tabletop applications also presents considerable technical challenges.

This dissertation addresses each of these areas. Firstly, a theory of workspace awareness is applied to consider the design of remote tabletop interfaces and the work practices that they may afford.

Secondly, two technical barriers to the rapid exploration of useful remote tabletop applications are identified: the low resolution of conventional tabletop displays; and the lack of support for existing user interface components. Techniques from multi-projector display walls are applied to address these problems. The resulting method is evaluated empirically and used to create a number of novel tabletop interfaces.

Thirdly, an empirical investigation compares remote and co-located tabletop interfaces. The findings show how the design of remote tabletop interfaces leads to collaborators having a high level of awareness of each other's actions in the workspace. This enables smooth transitions between individual and group work, together with anticipation and assistance, similar to co-located tabletop collaboration. However, remote tabletop collaborators use different coordination mechanisms from co-located collaborators. The results have implications for the design and future study of these interfaces.

Acknowledgements

I am indebted to many people for their support during this work. I thank Peter Robinson, my supervisor, for teaching me as an undergraduate, for inspiring me to enter research, and for guidance and encouragement throughout. I thank Alan Blackwell, my second adviser, for introducing me to the wider field and for patient guidance on all issues of validity. I also thank Mark Ashdown, whose own dissertation inspired and served as a guide early in the research.

Working in the Rainbow Research Group over the last four years has been a pleasure, and I owe much to my colleagues past and present for their insights, their patience, and, above all, their friendship. I thank my office mates Laurel Riek (and Riley!), Shazia Afzal, and those I shared with in FS18. Several people read drafts of this dissertation, and I thank Mark Ashdown, Alan Blackwell, Daniel Bernhardt, Neil Dodgson, Darren Edge, and Cecily Morrison for their insightful comments. I am also grateful to the innumerable people outside the group who have given useful feedback on this work. I particularly wish to acknowledge Fred Brooks, the CSCW'06 Doctoral Consortium, and the anonymous conference reviewers.

I owe much to the generous financial support of Thales Research and Technology (UK) and the EPSRC, without which I would not have had many of the opportunities I had. At Thales I particularly thank Chris Firth, Chris Ash, Terry O'Sullivan and Andy Vooght for their support and advice. Andy O'Byrne arranged installation of the projector rig in Guildford, and the Guildford research staff made my visits enjoyable. I also gratefully acknowledge the financial support of the Royal Academy of Engineering, the Cambridge-MIT Institute, the Cambridge Philosophical Society, the ACM, and Gonville and Caius College.

I am also most grateful to the many people who helped in other ways. Rowan Hill, Tom Matthews, Richard Russell, and Stephen Williams undertook undergraduate projects that I report in this dissertation. Graham Titmus provided patient guidance on all matters of equipment. Missy Cummings and her students at MIT's Humans and Automation Lab hosted my visit and made it enjoyable. Shazia Afzal, Daniel Bernhardt, Pradipta Biswas, Darren Edge, Lorcan Mac Manus, Cecily Morrison, Metin Sezgin, and Christian Richardt kindly posed for the photos that appear in this dissertation, while Ian Davies additionally loaned a camera and his skills as a photographer. Ian Burton-Palmer and Martin McDonnell arranged installation of the projector rig, while Carol Speed and Tanya Hall processed the endless travel receipts. The study volunteers generously gave their time and enthusiasm.

I thank all my friends, in the lab and elsewhere, for their encouragement. The Caius Grad VIII, though we didn't win blades, kept me motivated while writing up. I thank my family, and especially my parents and my sister, Fay, for encouraging me and educating me as far back as I can remember. Lastly I thank Emma, for supporting me through the highs and lows, and for making it all so much fun.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Effective support for synchronous remote collaboration has long proved a desirable yet elusive goal for computer technology. Consumer videophone technology was unveiled in 1964, amid forecasts of replacing standard telephony by the early 1970s (Egido, 1988). Forty years on, videophone technology is more easily available yet remains largely unused, and so perhaps little has changed in practice. Researchers have discussed a variety of problems, most notably poor reproduction of the visual cues, such as eye gaze, that mediate face-to-face conversation (e.g. Okada et al., 1994), and the inability to initiate and conduct the informal collaborative interactions that occur outside of formal scheduled collaboration (e.g. Bly et al., 1993). Both problems may soon be solved. The falling cost and commoditisation of large displays, camera equipment, and network bandwidth are making always-on remote "video windows" between spaces more feasible in practice, while novel techniques faithfully reproduce visual cues that were absent or distorted in previous technologies (Nguyen and Canny, 2005, 2007). These advances are currently making their way into commercial meeting-room systems.

Technologies that provide a shared visual workspace of the task at hand, rather than of the other remote collaborators, remain rather less advanced. Researchers have demonstrated large-format remote whiteboards to support fixed sketching (Tang, J. C. and Minneman, 1991a). However, remote collaboration for other workspace-based activities is largely confined to collaborative versions of conventional desktop-computer applications. These systems lack support for the visual cues and work practices that underpin visual workspaces in co-located collaboration, leading to well-documented problems (Gutwin and Greenberg, 2002).

Recent developments in technology to support co-located collaboration may offer opportunities to address these problems of remote work. The increasing size and resolution of affordable display technologies have led to predictions of ubiquitous

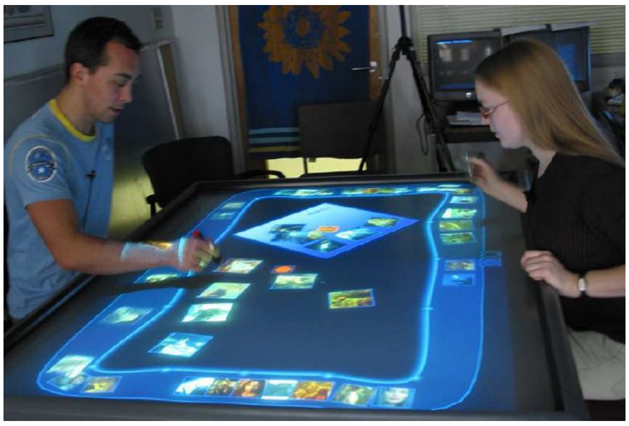


Figure 1.1: Example of a tabletop interface (from Hinrichs et al., 2005).

large displays (Weiser, 1999). Researchers investigating co-located collaboration have consequently constructed large horizontal interactive displays around which co-located collaborators can sit and interact using their hands. These *tabletop interfaces* present interactive shared task artefacts that appear on the display and mimic real-world task artefacts such as photos or puzzle pieces. Figure 1.1 shows an example.

Scott (2005) and Morris (2006) argue that traditional tables are prevalent in work environments because they are well-suited to many collaborative two-dimensional information tasks, such as planning, scheduling, brainstorming, design, and layout. Their size and orientation enables collaborators to sit around, and to spread out and spatially-organise the task artefacts. Tabletop interfaces aim to enable access to interactive content during such tasks, and in a way that affords established collaborative work practices. Tabletop interfaces have been demonstrated for tasks including planning (e.g. Rogers and Lindley, 2004) and design (e.g. Scott et al., 2005). They can afford beneficial work practices observed at conventional tables, such as fluid switching between individual and group work (Tang, A., et al., 2006b), and spatial partitioning as a coordination mechanism (Scott et al., 2005).

A tabletop approach might similarly offer benefits to shared workspaces for remote collaboration. A number of research projects have used large horizontal interactive surfaces to present a shared visual workspace to support remote collaboration (Wellner, 1993a; Ashdown and Robinson, 2005; Esenther and Ryall, 2006; Hutterer et al., 2006; Regenbrecht et al., 2006; Tang, A., et al., 2006a; Coldefy and dit Picard, 2007; Izadi et al., 2007). Figure 1.2 shows a number of these *remote tabletop interfaces*. The projects have been motivated by a range of factors. Some aim to replicate the affordances of tangible task artefacts, such as sheets of paper, on a large display (e.g. Ashdown and Robinson, 2005). Some are concerned with supporting group-to-group remote collaboration and postulate that sitting each group around a table will enable effective co-located collaboration as well as remote collaboration (e.g. Esenther and Ryall, 2006; Hutterer et al., 2006). Others postulate that adopting a tabletop approach in remote collaboration will result in the beneficial work practices, such as territorial use of space, that have been observed in co-located tabletop collaboration (e.g. Coldefy and dit Picard, 2007; Izadi et al., 2007).

The systems also differ in their design: some use indirect mouse input whereas others use direct stylus or touch input devices; some provide remote representations of collaborators' arms whereas others use telepointers or traces; some use interaction techniques developed for tabletop interfaces while others present legacy applications on a large horizontal display; and some use surfaces sufficiently large for collaborators to work side-by-side while others do not.

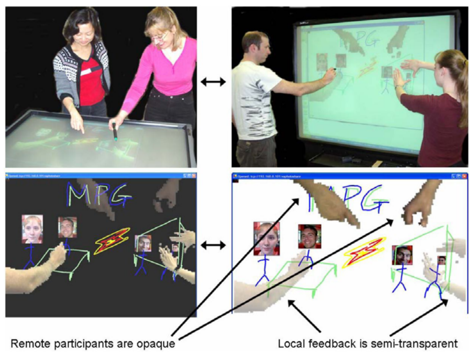
Above all, most of these projects have necessarily concerned themselves with the construction of the technology itself, which presents considerable challenges. There has consequently been little characterisation of the extent to which they support remote collaboration. Motivations of supporting collaboration have largely not been developed into well-grounded claims, and the interfaces have not been evaluated to determine whether such claims would hold in practice. Furthermore, the difficulties of constructing the technology are a barrier to the investigation of novel applications and interaction techniques. There has been little investigation, for instance, of remote tabletop interfaces to support document review meetings, collaborative web-browsing or data analysis. Interaction techniques for remote tabletop interfaces have been largely limited to those that have been demonstrated in co-located tabletop collaboration.

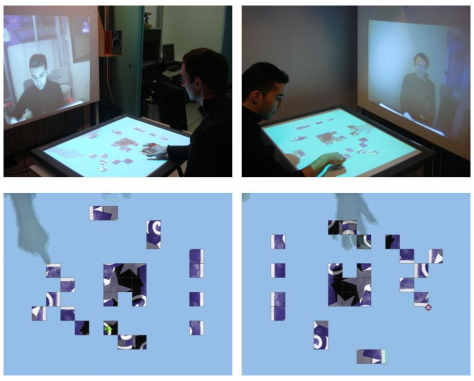
Accordingly, in spite of novel interfaces arising from interesting motivations and considerable technical work, several important challenges remain before the potential of remote tabletop interfaces can be determined. There is currently little theoretical basis for the design of remote tabletop interfaces, and little empirical characterisation of the extent to which they support remote collaboration. Rapid construction of remote tabletop applications and interaction techniques also presents considerable technical challenges. This dissertation addresses each of these areas.

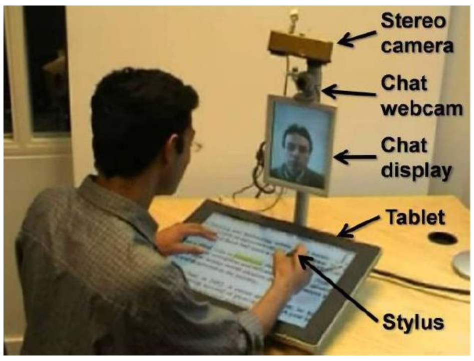
Having established the problem to be addressed, the remainder of this chapter provides an overview of the dissertation. It describes in turn the scope and goals of the research, the contributions towards knowledge in the field, and a chapter-

(a)

(b)


Remote participants are opaque
Local feedback is semi-transparent
(c)


(d)



(e)

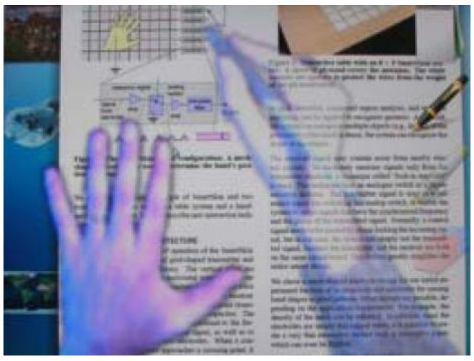

(f)

Figure 1.2: Examples of remote tabletop interfaces: (a) Escritoire (from Ashdown and Robinson, 2005); (b) Carpeno (from Regenbrecht et al., 2006); (c) VideoArms (from Tang, A., et al., 2006a); (d) DigiTable (from Coldefy and dit Picard, 2007); and (e and f) C-Slate (from Izadi et al., 2007).

by-chapter outline of the remainder of the dissertation.

1.1 Approach

This work falls within the field of Human-Computer Interaction (HCI), which is concerned with the impact of technology upon human behaviour and with how technology can be designed accordingly. My overarching goal throughout this dissertation is the study of remote tabletop collaboration in order to inform the future design of collaborative technologies. This is realised through three sub-goals:

Identification of theoretical motivations for the design of remote tabletop interfaces. After conducting a literature review of work practices observed in synchronous remote and co-located collaboration, I focused on support for workspace awareness as a primary motivation for the design. I applied a conceptual framework to postulate several mechanisms by which remote tabletop collaborators may maintain workspace awareness, and to develop design guidelines that may enable these mechanisms. The mechanisms, if they hold in practice, show that remote tabletop interfaces address some awareness shortcomings of conventional groupware systems. I also observed that collaborators using remote tabletop interfaces may exhibit several beneficial work practices associated with co-located tabletop collaboration.

Development of a method for the rapid construction of remote tabletop interfaces. From the literature review, I identified two technical barriers to the rapid exploration of novel remote tabletop applications: the low spatial resolution of typical current tabletop displays; and a lack of support for integrating existing user interface components with novel tabletop interaction techniques. I have explored a method to address these problems, investigating each sub-system in turn to review prior work, implement an appropriate solution and characterise its performance. The resulting system has been used by myself and by others to create a variety of novel tabletop interfaces for co-located and remote collaboration.

Investigation of work practices in remote tabletop collaboration. I conducted an exploratory study of remote and co-located tabletop collaboration to investigate some of the hypothesised awareness mechanisms and work practices. The study findings supported some of the investigated hypothesised awareness mechanisms and therefore supported some of the theoretical motivations and design guidelines. However, in spite of some similarities, the study identified that tabletop collaborators behave differently when remote than when co-located.

1.2 Contributions

This dissertation builds upon knowledge in the field to make three primary contributions:

Design of remote tabletop interfaces. Building on prior work, it uses workspace awareness to provide a theoretical basis for the remote tabletop approach. It describes several mechanisms by which remote tabletop collaborators maintain workspace awareness, and presents a set of design guidelines that, if followed, enable collaborators to use these mechanisms. Some of the guidelines and mechanisms are investigated and supported by the exploratory study. They inform the future design of applications and interaction techniques.

Construction of remote tabletop interfaces. The dissertation contributes a method for the construction of remote tabletop interfaces. The method provides for: a large shared workspace in which remote collaborators interact using tabletop techniques; segmentation of arms from a camera image of a front-projected display; increased spatial resolution in tabletop interfaces using a tiled array of projectors and techniques from multi-projector display walls; and a programming interface that enables rapid prototyping of novel tabletop applications through the reuse of existing user interface components. This novel combination of techniques enables the study of applications that previously were considered less feasible.

Work practices in remote tabletop collaboration. Lastly, the dissertation contributes findings about work practices in remote tabletop collaboration. These identify both similarities and differences in work practices between co-located and remote tabletop collaboration. They have implications for the design and future study of these interfaces.

1.3 Dissertation Structure

The structure of the remainder of this dissertation reflects the approach described earlier in this chapter.

Chapter 2 presents a review of the relevant literature to identify more thoroughly the research opportunities and provide context for the work that follows. It describes the evolution, motivations and design of co-located tabletop interfaces, and reviews work practices observed in co-located collaboration around conventional tables and interactive tabletop interfaces. It then describes the evolution, motivations and design of remote tabletop interfaces, highlighting differences between projects in their motivations and designs, and the few findings about the use of such interfaces in practice. Lastly, it reviews more general findings about language use and workspace awareness in shared visual workspaces for remote collaboration.

Chapter 3 presents a theoretical basis for the design of remote tabletop interfaces using a framework of workspace awareness reviewed in Chapter 2. This results in hypothesised awareness mechanisms and design guidelines. The chapter then discusses whether each of the work practices observed in co-located tabletop collaboration might apply in remote tabletop collaboration. The final part of the chapter again considers the literature reviewed in Chapter 2 to identify technical obstacles to the rapid construction of novel remote tabletop applications that meet the hypothesised guidelines.

Chapter 4 describes a method for the construction of remote tabletop interfaces that overcomes the difficulties identified in Chapter 3. It considers in turn the distribution architecture, display management, arm segmentation, programming interface, and reuse of existing user interface components. It presents, for each subsystem, the prior approaches and the chosen approach.

Chapter 5 presents an evaluation of the method presented in Chapter 4. It describes the evaluation of each subsystem in turn and presents several novel applications that illustrate the versatility and utility of the method. It discusses the immediate findings resulting from the evaluation and compares the method to other systems.

Chapter 6 describes an exploratory study investigating the awareness mechanisms, design guidelines and work practices hypothesised in Chapter 3. It describes the methods and results, and discusses the immediate findings.

Chapter 7 summarises the contributions of the dissertation. It discusses their wider implications for the future design and study of collaborative systems, and the new research opportunities presented.

1.4 Dissemination

The material in this dissertation also appears in a number of peer-reviewed papers.

Full papers:

Tuddenham, P., and Robinson, P. (2007). Distributed Tabletops: Supporting Remote and Mixed-Presence Tabletop Collaboration. In *Proc. TABLETOP'07: Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, pp. 19–26. (Chapters 3 and 4).

Tuddenham, P., and Robinson, P. (2007). T3: Rapid Prototyping of High-Resolution and Mixed-Presence Tabletop Applications. In *Proc. TABLETOP'07: Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, pp. 11–18. (Chapters 4 and 5).

Tuddenham, P., and Robinson, P. (2007). Improved Legibility of Text for Multi-projector Tiled Displays. In *Proc. PROCAMS'07: IEEE International Workshop on Projector-Camera Systems*, pp. 1–8. (Chapter 5).

Short work-in-progress papers:

Tuddenham, P. (2007). Distributed Tabletops: Territoriality and Orientation in Distributed Collaboration. Student Research Competition presentation. In *Ext. Abstr. CHI'07: ACM Conference on Human Factors in Computing Systems*.

Tuddenham, P., and Robinson, P. (2006). Remote Review Meetings on a Tabletop Interface. Doctoral Consortium presentation. In *Ext. Abstr. CSCW'06: ACM Conference on Computer-Supported Cooperative Work*.

Tuddenham, P., and Robinson, P. (2006). T3: A Toolkit for High-Resolution Tabletop Interfaces. In *Ext. Abstr. CSCW'06: ACM Conference on Computer-Supported Cooperative Work*.

Workshop position papers:

Tuddenham, P., and Robinson, P. (2007). Tabletop interfaces for remote document review meetings. Workshop on Collaborating over Paper and Digital Documents, London, November 2007.

Tuddenham, P., and Robinson, P. (2007). T3: Rapid prototyping of high-resolution tabletop applications. Shareable Interfaces Workshop, Milton Keynes, June 2007.

Tuddenham, P., and Robinson, P. (2006). Large displays for document-centric meetings. Workshop on Human-centred Technology, Brighton, England, September 2006.

Tuddenham, P., and Robinson, P. (2006). Large high-resolution displays for collaboration. Workshop on Information Visualization and Interaction Techniques, ACM Conference on Computer-Human Interaction (CHI), Montreal, Canada, April 2006.

Chapter 2

Background

This dissertation builds on work in the field of Computer-Supported Cooperative Work (CSCW), which is concerned with the design and study of technologies to support collaborative activities (Grudin and Poltrock, 1997). Such technologies are typically designed to support either synchronous collaboration (among collaborators working at the same time) or asynchronous collaboration (among collaborators working at different times), and to support either co-located or remote collaboration. Table 2.1 shows this widely-used time and space classification. In practice, collaborative work typically entails a variety of activities that include each of these collaboration categories, and so requires a combination of appropriate technologies.

	Asynchronous	**Synchronous**
Remote	E-mail	Telephone
Co-located	Notice board	Meeting table

Table 2.1: Classification of collaboration technologies.

The focus of this dissertation is the application of the tabletop approach, which has been developed and studied for synchronous co-located collaboration, to the problem of synchronous remote collaboration. This chapter presents a review of work in synchronous co-located and remote collaboration in turn, before concluding with a brief discussion.

2.1 Technologies for Co-located Collaboration

This section begins by describes tabletop interfaces for co-located collaboration, beginning with their evolution from earlier technologies. This provides a context for the review of tabletop interaction design and work practices in the sections that follow.

2.1.1 Single Display Groupware

Stewart et al. (1999) defined Single Display Groupware (SDG) as “computer programs that enable co-present users to collaborate via a shared computer with a single shared display and simultaneous use of multiple input devices”. This definition encompasses the vast majority of technologies designed to support co-located collaboration. Cognoter (Stefik et al., 1987b), part of the Xerox PARC Colab project, was an early example in which co-located collaborators sat at separate computers and manipulated a single shared wall display using separate mice. MMM (Bier and Freeman, 1991; Bier et al., 1992) enabled collaborators sitting at a single computer to use separate mice and keyboards to manipulate a shared editor. Later systems used alternative input devices such as handheld computers (Myers et al., 1998) and under-carpet weight sensors (Stanton et al., 2001).

These early systems addressed technical challenges, such as managing multiple input devices and replicating user interface components for each user (Bier and Freeman, 1991). Such techniques were later packaged as reusable toolkits (Tse and Greenberg, 2004). This early work also identified design challenges, such as sharing screen real estate between collaborators, directing feedback to one collaborator without distracting the other, and avoiding interference when collaborators are working separately (Bier and Freeman, 1991).

2.1.2 Direct Input Wall Interfaces

Early SDG systems were followed by vertical interfaces supporting direct stylus interaction for whiteboard-style sketching, such as Liveboard (Figure 2.1(a)) (Elrod et al., 1992) and Tivoli (Pedersen et al., 1993). These interfaces prompted the designers to reconsider the tacit assumptions of graphical user interfaces. They observed that direct input devices need no cursor and have greater scope for writing and sketching. The large space enables side-by-side collaboration. However, more effort is needed to move and reach across the display, and users must also step back if they wish to see the entire display. This necessitated more careful placement of on-screen controls and message windows. Unlike mice, the styluses and direct touch devices used in larger interfaces tended not have multiple buttons for context menus and mode selection, though they offered greater scope for gestural and bimanual input. More recent research has explored interaction techniques for managing space and history on large vertical interfaces, such as

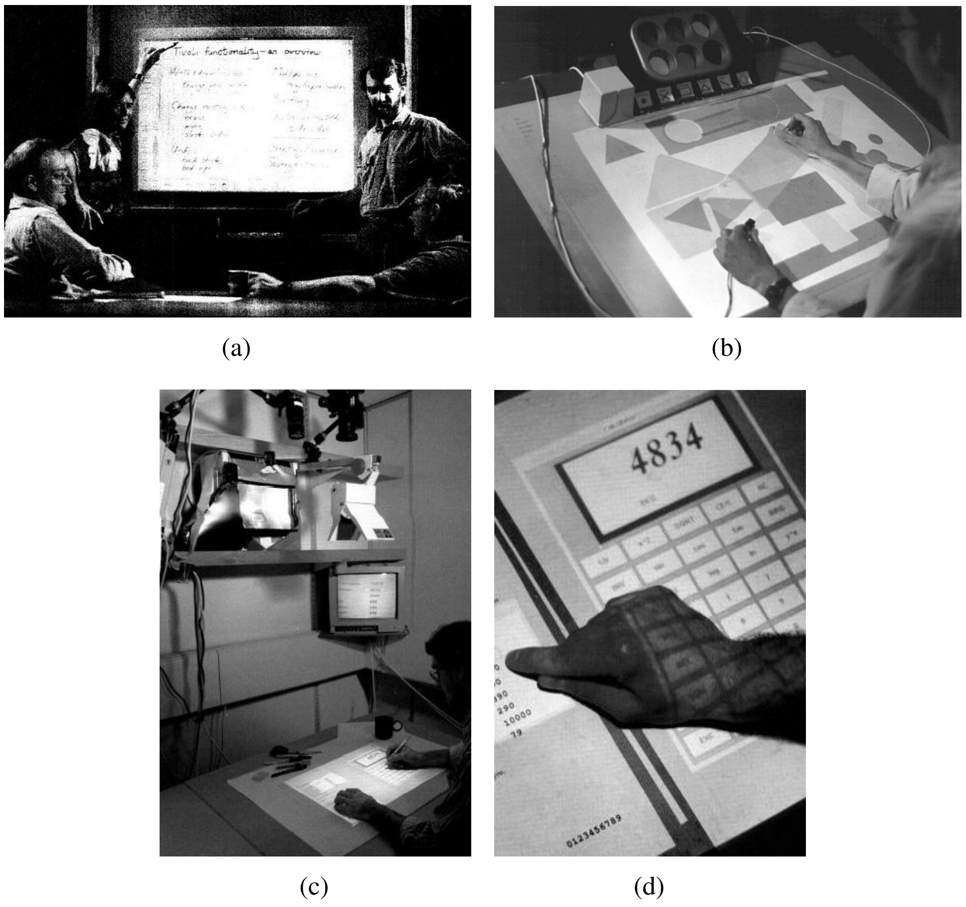


Figure 2.1: Examples of technologies for co-located collaboration: (a) Tivoli (from Pedersen et al., 1993); (b) Tangible Bricks (from Fitzmaurice et al., 1995); and (c and d) DigitalDesk (from Wellner, 1993a,b).

movable "sheets" of content (Mynatt et al., 1999; Guimbretière et al., 2001). Researchers have also begun to investigate the wider adoption of such interfaces into organisations (Huang et al., 2006).

2.1.3 Augmented Tangible Interaction Surfaces

A second class of large-format interfaces augments physical task artefacts on a horizontal surface with projected imagery. DigitalDesk (Wellner, 1993a,b) was an early project that augmented physical paper on a desk surface with interactive imagery with which a user could interact with a conventional pen (Figure 2.1(c

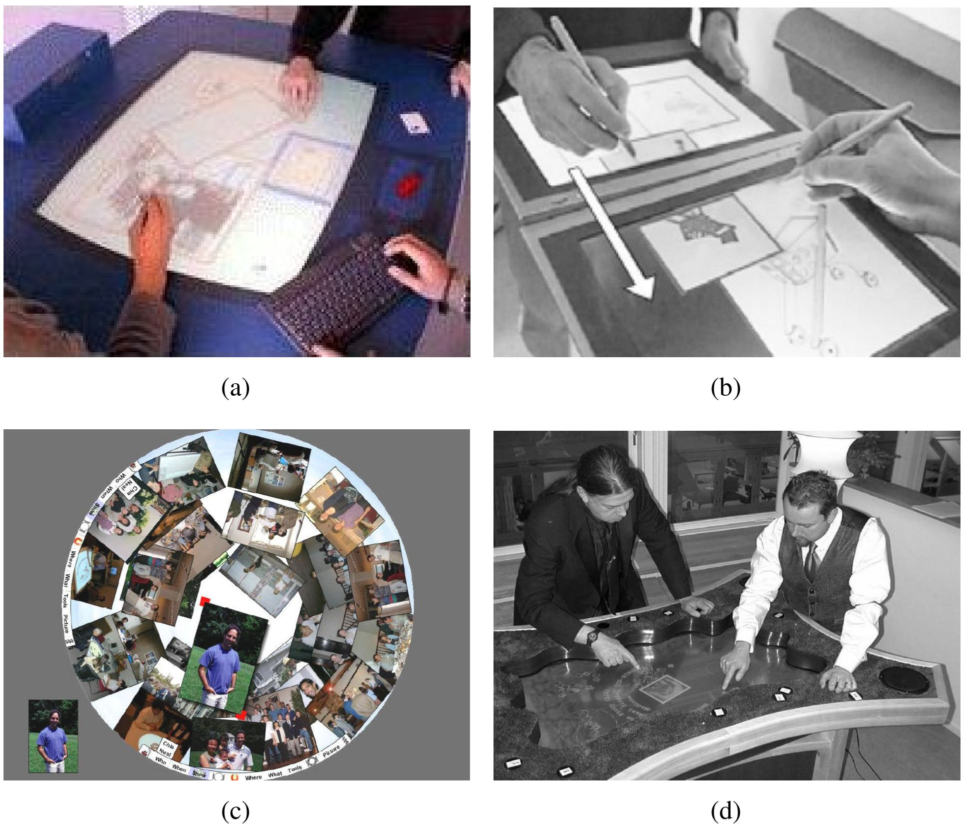


Figure 2.2: Early tabletop interfaces: (a) InteracTable (from Streitz et al., 1999); (b) Connectables (from Tandler et al., 2001); (c) Personal Digital Historian (from Shen et al., 2002); and (d) the Pond (from Ståhl et al., 2002).

and d)). Tangible User Interfaces (TUIs) similarly present augmented physical task artefacts such as blocks, typically on a surface. Users interact by physically manipulating the blocks themselves, for example by moving them (Figure 2.1(b)) (e.g. Fitzmaurice et al., 1995; Arias et al., 1997; Ullmer and Ishii, 1997; Rekimoto and Saitoh, 1999; Jordà, 2003). Various conceptual frameworks consider the design of tangible user interfaces from interactional (e.g. Edge and Blackwell, 2006) and social (e.g. Hornecker and Buur, 2006) perspectives.

2.1.4 Tabletop Interfaces

Augmented tangible interaction surfaces have been followed by large horizontal interactive surfaces on which the task artefacts are entirely non-physical and composed solely of computer-generated imagery. In the absence of physical task artefacts, collaborators instead use direct input devices such as multi-touch surfaces, sensor gloves and styluses. Several collaborators are typically able to interact concurrently. Such systems have been labelled *tabletop interfaces*.

The two-user Responsive Workbench (Agrawala et al., 1997) was one of the earliest such tabletop interfaces. Two collaborators wore sensor gloves and interacted with imagery projected onto the table surface. The imagery was made to appear three-dimensional by using shutter glasses and adjusting the image seen by each collaborator based on their head location. This was followed by other early tabletop interfaces (Figure 2.2) such as the InteracTable (Streitz et al., 1999), Connectables (Tandler et al., 2001), Personal Digital Historian (Shen et al., 2002) and the Pond (Ståhl et al., 2002). These later interfaces presented only two-dimensional task artefacts in order to avoid shutter glasses which, as Morris et al. (2004a) observe, inhibit eye gaze and so harm collaboration. Three-dimensional interactive surfaces have only recently been revisited (Grossman and Wigdor, 2007).

Since these early systems, there has been an explosion in tabletop interface research (e.g. Fjeld and Takatsuka, 2006; Ryall and Scott, 2007). Tabletop interfaces have been demonstrated for a variety of group activities, such as creating personal histories of photos (Shen et al., 2002), sketching (Morris et al., 2006a), story-telling with photos (Apted et al., 2005; Mazalek et al., 2007), design tasks (Scott et al., 2005), data exploration (Forlines and Shen, 2005), urban planning (Tang, A., et al., 2006b), map exploration (Tse et al., 2006), search (Ståhl et al., 2002), board games (Piper et al., 2006) and video games (Tse et al., 2006). Microsoft have also recently unveiled their Surface Computing project to commercialise tabletop interfaces.

As described in Chapter 1, conventional tables are well-suited to a wide variety of collaborative two-dimensional information tasks in which task artefacts are spread out and spatially arranged (Scott, 2005; Morris, 2006). Tabletop interfaces aim to enable similar collaboration using interactive content, while supporting established collaborative work practices. For this reason, much tabletop interface research has investigated interaction techniques and work practices in such layout tasks. This material is reviewed in the following two sections, followed by a brief overview of tabletop hardware configurations.

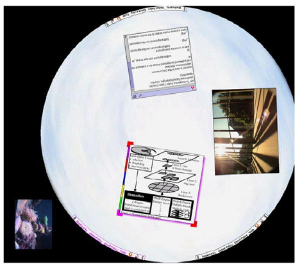
2.2 Interaction Design for Co-located Tabletop Collaboration

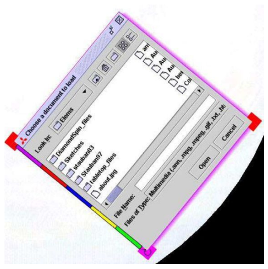
Conventional graphical user interface techniques are designed for a single user with a small vertical screen, mouse and keyboard, but tend to be inappropriate for tabletop interfaces. Like large direct-input wall interfaces, tabletop interfaces require careful placement of user interface controls to avoid strenuous reaching. Similarly, direct input devices often lack buttons but offer greater scope for gestural and bimanual techniques. Tabletop interfaces introduce further problems because people interact around the table from different sides, whereas conventional user interfaces use a single orientation for the presentation of text and placement of controls. As Scott (2005) observes, tabletop interfaces accordingly present both challenges and opportunities for new interaction design.

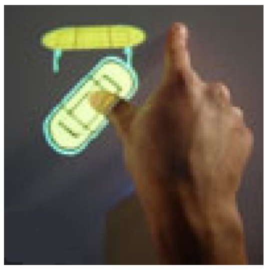
2.2.1 Movable Task Artefacts

From the earliest projects onwards, many tabletop interfaces have presented users with small virtual task artefacts, such as photos, that they could move and rotate on the surface. Streitz et al. (1999), for instance, reported “We developed gestures for rotating and shuffling individual and groups of information objects across the surface. This accommodates easy viewing from all perspectives”. Tandler et al. (2001) describe how objects on Connectables can be “moved... and rotated, just as one would do with a sheet of paper” and that “the rotation of objects allows adapting the alignment of documents to the positions of the people standing around the table.” It remains unclear whether these researchers believed small movable task artefacts to be implicit in the need to support different orientations, or whether they were motivated by earlier tangible systems. Either way, this approach is very different from conventional desktop user interfaces, which tend to use rigid, automatic arrangement in a scrollable window.

This practice of presenting small movable task artefacts where possible has become commonplace in many, though not all, tabletop interfaces (e.g. Shen et al., 2002, 2004; Ryall et al., 2004; Hinrichs et al., 2005; Scott et al., 2005; Isenberg et al., 2006; Kruger et al., 2005; Apted et al., 2006). Interaction with such virtual task artefacts inevitably remains very different to interaction with physical task artefacts (Terrenghi et al., 2007). However, tabletop interaction seeks not necessarily to mimic physical task artefacts but rather to afford the same work practices observed at conventional tables.


(a)


(b)


(c)

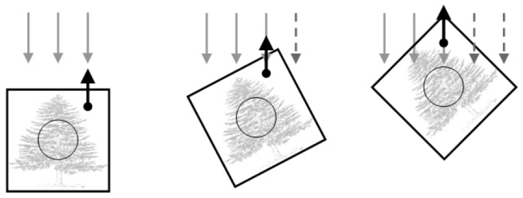

(d)

Figure 2.3: Examples of interaction techniques for orienting task artefacts: (a) automatic outwards orientation (from Shen et al., 2004); (b) rotation using handles (from Shen et al., 2004); (c) rotation using gestures (from Wu and Balakrishnan, 2003); and (d) the Rotate'N'Translate technique (from Kruger et al., 2005).

2.2.2 Task Artefact Orientation

Kruger et al. (2004) review techniques for orienting task artefacts on a horizontal surface. Some systems automatically align task artefacts outwards or towards the nearest table edge, enabling passing between collaborators (Figure 2.3(a)) (e.g. Shen et al., 2004). The Lumisight table (Matsushita et al., 2004) used a novel construction of multiple projectors that enabled each collaborator sitting around the table to be presented with their own personalized view of the workspace. Such a system would enable every collaborator to see every task artefact oriented towards themselves. The different views presented to each collaborator may, however, disrupt collaborative processes, such as gesturing, that rely on a shared view.

The orientation of task artefacts not only affects the comprehension of text, but also is a mechanism for coordination and communication. Collaborators turn artefacts towards each other, for instance, to attract attention and to establish an audience (Kruger et al., 2004). (These findings are reviewed in the following section.) The above automatic orientation techniques may therefore be detrimental to collaboration, and several alternative techniques allow manual rotation of task

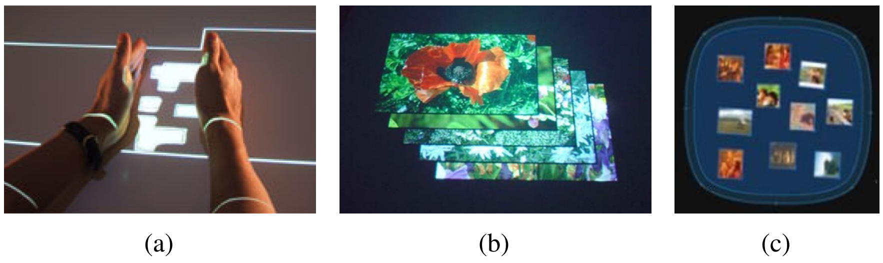


Figure 2.4: Examples of interaction techniques for grouping task artefacts: (a) using gestures (from Wu and Balakrishnan, 2003); (b) piling (from Ashdown and Robinson, 2005); and (c) shrinking task artefacts into storage container task artefacts (from Scott et al., 2005).

artefacts to arbitrary orientations. One technique uses a reorient icon in the corner of each task artefact, rather like conventional desktop drawing applications (Figure 2.3(b)) (e.g. Streitz et al., 1999; Tandler et al., 2001; Shen et al., 2004). Others use two-fingered or bimanual gestures for rotation (Figure 2.3(c)) (Wu and Balakrishnan, 2003). Finally, Rotate'N'Translate (Kruger et al., 2005) is a widely-adopted technique that uses a model based on physics to enable simultaneous rotation and translation of a task artefact using a single continuous stylus stroke or finger action (Figure 2.3(d)).

2.2.3 Task Artefact Grouping

Various interaction techniques address the problems of forming groups of task artefacts on tabletop interfaces, moving groups of task artefacts, and visualisations for large groups. Such techniques are motivated by a study of office workers showing that piles of task artefacts provide a cognitively lightweight storage mechanism without distracting from the task at hand (Malone, 1983). Movable task artefacts on tabletop interfaces can be arranged in groups simply by moving them closer together. Interfaces that further support overlapping of task artefacts enable creation of piles. Wu and Balakrishnan (2003) demonstrated hand gestures for moving items into a pile (Figure 2.4(a)), and Ashdown (2004) demonstrated piles that can be temporarily expanded for viewing and manipulation of individual task artefacts (Figure 2.4(b)).

Collaborators at a conventional table move piles around the table both to minimise reaching and also to indicate the availability of task artefacts (Scott et al., 2004). (These findings are reviewed further in the following section.) Scott et al.

(2005) aimed to support such behaviour in tabletop interfaces by providing movable storage container task artefacts into which other task artefacts can be moved (Figure 2.4(c)). When a storage container is moved, any contained task artefacts move with it. Other tabletop interfaces have also demonstrated movable grouping mechanisms (Ashdown and Robinson, 2005; Hinrichs et al., 2005).

Task artefacts can be shrunk to enable large numbers to be stored without overlap or occlusion, which is detrimental to visual search. Task artefacts are typically scaled to smaller sizes when they enter peripheral areas of the display (Guimbretière et al., 2001; Shen et al., 2004) or storage containers (Scott et al., 2005; Apted et al., 2005).

2.2.4 Individual and Group Work

Tabletop interfaces can support both group work, which tends to take place in the centre of the table, and individual work, which tends to take place immediately in front of each collaborator (Tang, J. C., 1991; Kruger et al., 2004; Scott et al., 2004; Tang, A., et al., 2006b). (These findings are reviewed in the following section.) This has motivated a range of techniques that aim to support individual and group work.

Some interfaces have demonstrated explicit private workspaces into which task artefacts can be moved for individual work. A tabletop interface may be augmented with extra "private" display devices such as laptops, onto which task artefacts can be moved using novel dragging techniques (Rekimoto, 1998; Shen et al., 2003). These extra devices provide higher resolution appropriate for individual work, but remaining aware of others' activities becomes more difficult. Collaborators may alternatively be presented with individual views of the table using shutter-glasses (Shoemaker and Inkpen, 2001) or multiple projectors (Matsushita et al., 2004), though this is likely to disrupt processes such as gesturing, since collaborators no longer share a common view. Tabletop interfaces can, in any case, successfully support individual work without requiring such interventions as extra displays or individual views (Scott et al., 2005; Tang, A., et al., 2006b).

Ringel et al. (2004) proposed an explicit personal access control in which a task artefact could be manipulated only by its owner. They investigated various techniques for transitioning a task artefact from personal to public access control, such as rotating it towards the group, moving it to the centre of the table, enlarging it, and otherwise marking it as accessible ("releasing"). Moving to the centre of the table was easiest, fastest, and resulted in fewest errors. However, Scott (2005) observed that personal access controls place a heavier burden on the owners of task artefacts to relinquish ownership explicitly than is ordinarily required by social protocol. Again, tabletop interfaces can, in any case, successfully support individual work without requiring such access control interventions, by relying on social protocol alone (Scott et al., 2005; Tang, A., et al., 2006b).

Access controls are perhaps more important when considering global actions that affect the whole group, such as switching the visualisation mode of the entire display, spinning the table, clearing group work, or automatically gathering all task artefacts into a pile. Morris et al. (2004b) proposed a variety of access control policies, such as voting. They further proposed cooperative gestures, in which collaborators must all perform the same gesture simultaneously to instigate a global action (Morris et al., 2006a), and also that global actions be instigated using a single centrally-located set of controls (Morris et al., 2006b). The switching of visualisation mode can be avoided as a global action by instead using small movable lenses (Bier et al., 1993) whose visualisation modes can be switched without affecting the rest of the table (Forlines and Shen, 2005; Tang, A., et al., 2006b). Tang, A., et al. (2006b) found that collaborators using the lenses spent a greater proportion of time performing individual work than those using a global visualisation mode switch.

2.2.5 Summary

In summary, tabletop interfaces have required different interaction design from conventional graphical user interfaces. They tend to present task artefacts that collaborators can freely move, reorient and regroup on the surface using direct input mechanisms such as hand or stylus gestures. Interventions such as extra displays and access controls are thought to be unnecessary for supporting individual work at tabletop interfaces. Interventions such as access controls, cooperative gestures, or careful placement of controls, may be important for mediating global actions that affect the entire workspace.

2.3 Work Practices in Co-located Tabletop Collaboration

This section reviews three work practices at conventional tables and at interactive tabletop interfaces: switching between individual and group work; spatial partitioning of the workspace; and the use of orientation of task artefacts.

The review draws on a number of observational studies of tabletop collaboration. Tang, J. C. (1991) conducted an observational study of teams of three and four collaborating around a table on a design task using pens and paper sheets. Scott et al. (2004) conducted an observational study of groups of two and three collaborating around a conventional table on a furniture layout design task using paper and sketches to represent furniture. Rogers and Lindley (2004) carried out observational studies of teams of three performing a decision-making and planning task in three conditions: an interactive tabletop interface; an interactive wall alongside a coffee table on which they could make paper notes; and a conventional desktop PC. Rogers et al. (2004) observed teams of three performing decision-making tasks using a tabletop interface. Scott et al. (2005) developed a tabletop interface with a mobile container task artefact and observed its use in a task involving sorting and arranging photos. Kruger et al. (2004) conducted an observational study of orientation of task artefacts in a two-person word puzzle game on a conventional table. Finally, Ryall et al. (2004) examined the effects of group size and table size in a word puzzle task using a tabletop interface with groups of two, three and four collaborators.

2.3.1 Supporting Group and Individual Work

Collaborators at conventional tables are able to transition rapidly and fluidly between working closely as a group and working individually on part of the task. Tang, J. C. (1991), for instance, observed that some paper sketches were created to present ideas to the group, whereas others were created while working individually. Scott et al. (2004) reported that individuals would temporarily disengage from the group activity to perform individual activities, such as investigating different solutions to the group task. Kruger et al. (2004) described how gestures and manipulations conducted during individual work were ignored by other team members, whereas when interacting with the group, collaborators established an audience for their talk and gestures. More generally, Salvador et al. (1996) described *coupling* as the dependency of one collaborator on another. *Loosely-coupled* collaborators require relatively few interactions with each other to make progress, whereas *tightly-coupled* collaborators must interact with each other frequently.

Two studies have observed transitioning between a variety of coupling styles around tabletop interfaces. Tang, A., et al. (2006b) conducted an observational study of groups of two performing a map-based route-planning task that required visualisation of mutually-occluding layers of data. The study compared a single global control for selecting layers against a lens-based approach in which each participant had a movable “lens” task artefact (Section 2.2.4). They observed not a simple dichotomy but rather a variety of coupling styles, ranging from working together on the same problem in the same area, to closely watching each other manipulate the display, to working simultaneously in different areas on the same problem, watching each other work, and finally working completely independently on different problems. The interaction technique (lenses or global) influenced the proportion of time spent in each coupling style. Scott et al. (2005) observed that pairs divided the photo-arranging task into two stages. The first stage, of coarse-grained organisation of the photos, was conducted working together. The second stage, of fine-grained organisation, was conducted either working together or working individually using a divide-and-conquer strategy, depending on the pair.

Rogers and Lindley (2004), in their comparison of interfaces, supported only a single stylus (and therefore only a single collaborator) interacting at any time. Accordingly they report "little evidence of parallel or separated working" at the tabletop interface. They do, however, report that the tabletop interface condition resulted in more frequent switching of the role of interactor between collaborators than the wall display or the desktop PC. They attribute this to the interactor laying down the stylus on the table in order to gesture, which then invited one of the other collaborators to pick it up and take on the role. They further report that collaborators found it easiest to switch between making individual paper notes and interacting with the shared display when using the tabletop interface.

Rogers et al. (2007) similarly investigated equitable participation in an observational study of groups undertaking a collaborative design task in three conditions: a laptop; a multi-touch tabletop interface; and an augmented tangible interaction surface that also used a multi-touch input device. The large horizontal surface conditions led to more equitable participation than the laptop condition, both in interactions with the system and utterances to each other. They suggest that this arises because such surfaces offer "more equal opportunities for all collaborators to physically take part in the task".

2.3.2 Spatial Partitioning

A number of studies have shown that collaborators at conventional tables partition the workspace into regions. Tang, J. C. (1991) observed that collaborators would draw sketches in relation to existing drawings to establish an audience, whereas sketches drawn small and in the region immediately in front of a collaborator were "within a personal boundary and not intended for others to perceive". Scott et al. (2004) further investigated this behaviour using the furniture layout task. They observed that collaborators tacitly divided the workspace into regions that were subject to different social norms, which they labelled territories. Personal territory, the area immediately in front of a collaborator, was reserved for their own use, as an area into which they could move task artefacts to reserve them for themselves, and in which they could perform actions away from the group. By moving task artefacts to the personal territory, collaborators could also perform activities such as reading, writing and sketching without the ergonomic strain of reaching. The size and location of personal territories depended on the number of collaborators and seating arrangement, the table size, the task activities, the space occupied by the task artefacts, and visible barriers such as the edge of the floor plan. The table area not covered by the personal territories was considered group territory, a space in which main task activities were conducted and into which task artefacts could be moved to indicate their availability for group use. The authors reported that group space in the furniture layout task was tacitly partitioned so that each collaborator took responsibility for the nearest region. Finally, storage territories were table regions in which collaborators stored and organised groups of task artefacts not currently in use. Collaborators would move the piles around the table to access them more easily and to indicate their availability for group use. In summary, territoriality serves various roles to aid coordination in collaboration at conventional tables.

Tabletop interfaces at which collaborators sit and interact with movable task artefacts might be expected to yield such territorial behaviour. However, the tabletop interfaces demonstrated to date do not require collaborators to move task artefacts to avoid ergonomic strain: they do not provide sufficiently high display resolution to support reading of small text; and nor do they require intricate manipulation such as writing or using scissors. It is therefore not clear whether tabletop interfaces will support territorial behaviour matching the rich descriptions of the above studies. Two studies have reported limited territorial behaviour on interactive tabletop interfaces. Scott et al. (2005) developed a mobile container task artefact and observed its use in the photo-arranging task. They observed that collaborators working together would position the container centrally so that they could both access it, whereas when working individually collaborators would move containers closer to their own page for side-by-side comparison. Ryall et al. (2004), studying the interactive word puzzle task, logged and plotted the locations of each collaborator's interactions. They found that collaborators tended to interact in the nearest region of the table and, though working together, were reluctant to manipulate words near another group member.

2.3.3 Orientation of Task Artefacts

Tang, J. C. (1991) observed that the spatial orientation of sketches with respect to collaborators around the table was not only a problem (because of the lack of a shared common viewpoint) but also a resource. He observed that sketches would be created to face a particular collaborator in order to establish an audience, whereas sketches drawn facing oneself were considered individual activity. Kruger et al. (2004), in their study of collaboration at a conventional table, found that orientation served three roles. Firstly, most obviously, collaborators oriented task artefacts for comprehension, to enable themselves and others to read and manipulate, and to gain an alternative perspective on the task. Secondly, as Tang, J. C. (1991) observed, orientation served as a coordination mechanism to establish spaces and task artefact ownership. Collaborators tended to orient task artefacts towards themselves in the area nearest to them, whereas central task artefacts were typically oriented to a common group orientation. Collaborators would indicate availability when placing task artefacts on the table, by orienting them either towards themselves or towards the group. Lastly, orientation is a mechanism for intentional communication. Collaborators would orient task artefacts either towards themselves, towards another collaborator, or towards the group, in order to establish an audience for their talk and gestures.

In spite of various interaction techniques for reorienting task artefacts on tabletop interfaces (Section 2.2), surprisingly few studies have investigated this work practice. Rogers et al. (2004) observed that collaborators would rotate task artefacts to the group orientation in order to invite discussion. This supports the use of orientation as a communication mechanism.

2.3.4 Summary

In summary, collaborators at tabletop interfaces can exhibit beneficial work practices that have been observed at conventional tables. In particular, they can transition fluidly between individual and group work, and use spatial partitioning and task-artefact orientation as coordination mechanisms.

2.4 Tabletop Hardware Configurations

Tabletop interfaces are typically constructed using a single desktop PC linked to a single conventional rear- or front-mounted projector to create an image on the display surface. Projectors have generally been preferred over other technologies such as LCD panels and plasma screens because they offer greater scope to create larger displays. Front projection results in shadows cast by the hands and arms on the display, though in practice these fall very close to the hands and are not problematic. Ashdown (2004) projects at a slightly oblique angle to mitigate the effect of shadows when users lean over the display from one side.

Tabletop interfaces and wall displays have demonstrated a variety of direct input technologies:

Commercially-available stylus systems. Mimio, an ultrasonic stylus, is relatively inaccurate and tracks only a single stylus at a time (Ashdown, 2004). Large tablet systems, such as GTCO CalComp, provide more accurate tracking of a single stylus (Ashdown, 2004). Polhemus Fastrak tracks multiple tethered styli with six degrees of freedom, enabling novel techniques for interacting with task artefacts at a distance (Parker et al., 2005). Anoto styluses use a novel tracking mechanism in which a camera located in the stylus nib detects the stylus position on a surface printed with a special dot pattern. This provides high accuracy and precision (`http://www.anoto.com/`). The coordinates are streamed out over Bluetooth, and multiple styluses can be used concurrently. Haller et al. (2006) demonstrated a front-projected tabletop interface using this technology. Stylus systems typically allow users to lean on the table and to place physical objects on the table while interacting, as they might with a conventional table, whereas the bare-hand input technologies described below typically do not.

Commercially-available touch-sensitive overlays. Conventional overlays, used in kiosks, do not support multiple simultaneous touches. SMARTBoard DViT overlays, designed for wall displays, use infra-red LEDs and cameras mounted at the table edge to provide sensing over a large area of up to two simultaneous touches (Morrison, 2005).

Capacitive sensing surfaces such as DiamondTouch (Dietz and Leigh, 2001) and SmartSkin (Rekimoto, 2002) were designed specifically for tabletop interaction. A grid of capacitive sensing antennas inside the surface senses multiple simultaneous touches. DiamondTouch additionally identifies each touch with the corresponding user, by tethering each user to a pad that couples a weak electric signal through their body. DiamondTouch is commercially available and has been widely used.

Commodity cameras and computer vision techniques. Frustrated total internal reflection techniques use LEDs mounted at the edge of an acrylic display surface, and a rear-mounted camera (Han, 2005). The light from the LEDs remains within the acrylic because of total internal reflection, but escapes towards the camera at points where a finger is in contact. An alternative technique uses a rear-mounted infra-red light source, and a rear-mounted camera with an infra-red-only filter. Light from the source is reflected back towards the camera at points of contact (Matsushita and Rekimoto, 1997).

Wilson (2005) demonstrated a novel system using a front-mounted projector, a front-mounted infra-red light source and a front-mounted camera with an infra-red-only filter. Touches are inferred from the shape of the shadow cast by the fingers in the infra-red light.

Stereo cameras combined with computer vision techniques can infer depth sufficiently accurately to act as a touch-sensitive surface when using rear projection (Agarwal et al., 2007).

2.5 Classification of Remote Collaboration Technologies

I now consider *synchronous remote collaboration technologies*, which augment an audio channel with visual information to enable communication between geographically-separated collaborators. These can be divided into broadly two classes (Buxton, 1992): *person-space technologies* present video of the other person's face and body; while *task-space technologies* present a shared visual workspace in the domain of the task. Although some research technologies and commercial products provide both person-space and task-space, the characterisation of desirable attributes and the development and evaluation of technologies have progressed separately in the two distinct classes.

Collaborative virtual environments (CVEs) are a notable exception to this classification. They build on virtual reality technologies, which immerse a single user or co-located group into a virtual three-dimensional world in which they can typically move and interact with virtual three-dimensional task artefacts. CVEs present a three-dimensional world in which remote collaborators are represented by avatars (e.g. Krueger et al., 1985; Krueger, 1993; Greenhalgh and Benford, 1995; Fraser, 2000; Hindmarsh et al., 2000; Regenbrecht et al., 2002; Sonnenwald et al., 2004). The technologies vary in fidelity in various ways, such as whether or not the face of the avatar is a video image of the face of the collaborator, whether the movements of the avatar are controlled by the those of the collaborator using a tracker system or by the keyboard and mouse, and the extent to which the display technology immerses the collaborator in the system. CVEs offer the opportunity to transport collaborators into an entirely artificial three-dimensional world. This may be advantageous when collaborating over task artefacts that are inherently three-dimensional, such as a product design, or when using three-dimensional space to explore a virtual world, such as Second Life (Rymaszewski et al., 2006). However, this is offset by the fidelity limitations of current technology. It is unlikely, for example, that current CVEs could convey the rich hand gestures and sketching activity that can be demonstrated using remote task-space collaboration technologies.

2.6 Remote Collaboration in Person Space

Given the focus of this dissertation, I provide only a very brief review of person-space remote collaboration (defined in Section 2.5 above). Finn et al. (1997) provide thorough coverage of much of this material. Egido (1988) reviews early efforts to provide bidirectional video communication showing each collaborator's head and shoulders, and describes how the low adoption of early consumer videophone technology led researchers to focus instead on videoconferencing systems to avoid travel when conducting business meetings, which achieved gradual adoption.

Whittaker (1995) reviews the early controlled experiments studying the effects of communication media such as video, audio-only and face-to-face (e.g. Short et al., 1976). He identifies three claims made about the ways in which head-and-shoulders video supports the non-verbal aspects of communication, and summarises the experimental evidence for each:

1. Cognitive cues, such as head nods, facial expressions, and posture, enable collaborators to monitor each other's understanding and awareness
2. Turn-taking cues, such as eye gaze and body orientation, enable collaborators to manage floor control by indicating a desire to take or hold the floor.
3. Social cues, such as eye gaze, facial expressions, and body language, increase collaborators' interpersonal awareness by enabling them to judge reactions, agreement and emotions.

Whittaker's review argues that only the third claim is substantially supported by evidence. More recent work in this area has shown that person-space video offers benefits when collaborators can gesture to "sketch out" ideas in the air to each other (Veinott et al., 1999). Studies also suggest that visual cognitive cues are important in larger groups because it becomes more difficult to use verbal back-channels (such as saying "uh-huh"), and to resolve individual misunderstandings (e.g. Chen, M., 2002).

Real-world deployments are also different from laboratory experimental equipment, because compression techniques applied to the video signal introduce latency. In such systems the audio is either delayed to match the video, which severely impacts floor management, or transmitted without synchronisation with the video, which affects a collaborator's ability to lip-read (Chen, M., 2003).

There are two further research themes within person-space technologies. Media spaces (Stults, 1986) refers to several projects that investigated the role of video in initiating informal gatherings, maintaining working relationships and providing availability cues, rather than meeting support. These investigations tended to involve long-term field deployments of large permanent "windows" between public spaces (Fish et al., 1990; Jancke et al., 2001; Karahalios and Donath, 2004) and configurable links between offices (Mantei et al., 1991; Gaver et al., 1992; Fish et al., 1992; Bly et al., 1993; Hindus et al., 1996).

A second theme of research addresses the problem of conveying spatial cues in conventional person-space video technologies. Such technologies typically distort mutual eye gaze (eye contact) and gaze awareness (awareness of the current object of a collaborator's visual attention), and similarly do not convey faithfully the direction of body posture and gesture. These problems arise because the camera at each site is typically situated to one side of the display, and so when a collaborator attempts mutual eye gaze, the camera image shows them looking away. The problem is more complex in group-to-group collaboration where each site has only a single display and camera. Various technologies have addressed these problems (e.g. Okada et al., 1994; Sellen, 1995; Morikawa and Maesako, 1998; Raskar et al., 1998; Vertegaal et al., 2003). Most recently, Nguyen and Canny (2005) demonstrated a novel system that addresses these problems in group-to-group collaboration. They use multiple rear-mounted cameras and a large novel screen that can show a different image to each collaborator depending on their position at the table. They later empirically showed that distorted spatial cues negatively affect trust-formation between remote groups, and that their system eliminates this problem (Nguyen and Canny, 2007).

2.7 Remote Collaboration in Task Space

Early task-space technologies for remote collaboration presented a shared visual workspace on a conventional screen using a conventional keyboard, mouse and graphical user interface environment (e.g. Pferd et al., 1979). Toolkits such as GroupKit (Roseman and Greenberg, 1996) later provided software infrastructure to create such systems, including support for multiple telepointers and concurrent interaction by multiple users. More recently, software such as Microsoft Netmeeting and Google Docs have brought such systems to a wider audience. Such systems are sometimes called groupware, and for clarity in this dissertation I shall refer to them as *conventional groupware*.

This section reviews the evolution of tabletop interfaces for remote collaboration. It begins by reviewing the departure from conventional groupware towards larger-format interfaces for remote collaboration, starting with the studies of collaborative design activity that motivated the change. It then describes several remote tabletop interfaces and reviews observations about their use in practice.

2.7.1 Studies of Collaborative Design

The move from conventional groupware towards larger-format remote collaboration interfaces was motivated partly by the findings of observational studies of co-located collaborative design work.

Following an earlier study by Bly (1988), Tang, J. C. (1991) studied the use of shared drawings in design meetings on conventional tables. In addition to the observations in Section 2.3, he also reported several other findings: hand gestures not only express ideas but also mediate interaction by negotiating turn-taking and focussing attention, and are imitated by other collaborators when discussing ideas; the process of creating and discussing drawings is frequently more important than the resulting drawings themselves; and drawing, writing and gesturing are fluidly interleaved.

Tang's work was followed by several further studies of co-located collaborative design (e.g. Olson et al., 1992). Bekker et al. (1995) analysed data from these studies in order to generate design guidelines for collaboration technology. They identify four classes of gesture: *point gestures* for deixis; *spatial gestures* indicating size, distance or relative location; *kinetic gestures* whose movement indicates an action; and gestures that do not fit the other categories, such as to emphasise, to facilitate speech production or to attract attention. Like Tang, J. C. (1991) they observed that gestures were brief and interleaved with other activities such as drawing. They also observed that gestures sometimes referred to previous gestures, such as places where earlier gestures were made; that gestures sometimes mimicked earlier gestures; and that gestures sometimes made reference to imaginary objects created by earlier gestures.

2.7.2 Large-Format Remote Sketching

These studies of collaborative design motivated a shift in remote collaboration technologies away from conventional groupware and monitor and mouse interaction. Researchers presented a number of larger direct-input remote collaboration interfaces that were designed to support the behaviours observed in these studies,

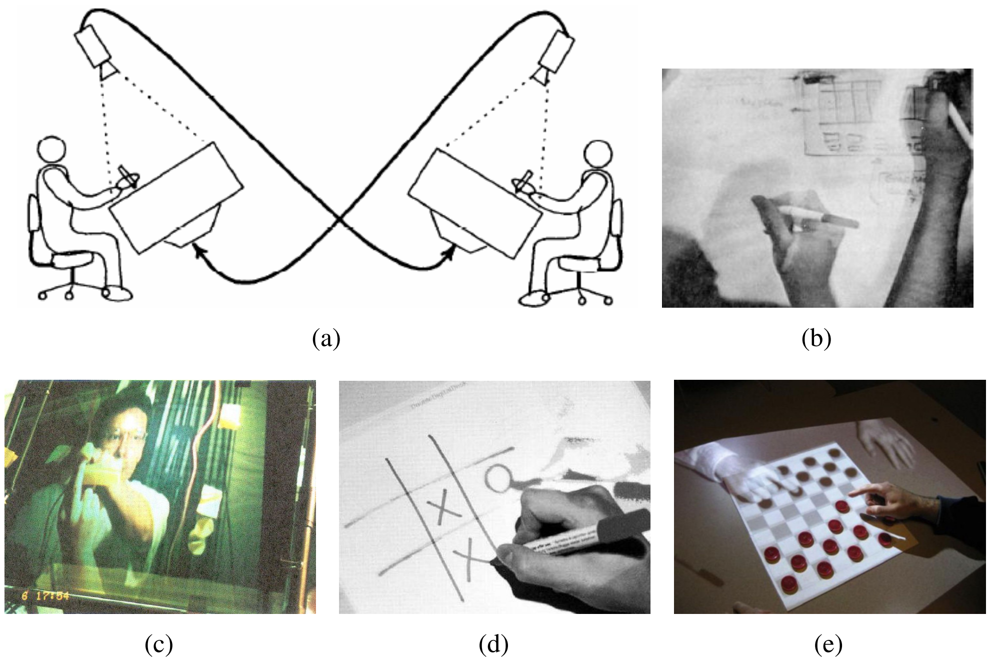


Figure 2.5: Examples of remote collaboration technologies: (a and b) VideoDraw (from Tang, J. C. and Minneman, 1991b); (c) ClearBoard (from Ishii and Kobayashi, 1992); (d) Double DigitalDesk (from Wellner and Freeman, 1993); and (e) PlayTogether (from Wilson and Robbins, 2006).

such as interleaved sketching and gesturing, unmediated remote representation of hand gestures, and space to mediate interaction.

Commune (Bly and Minneman, 1990) and VideoDraw (Tang, J. C. and Minneman, 1991b) both linked horizontally-mounted monitors and supported simultaneous sketching using styluses, gesturing, and instant visual updates to all sites as users sketched (Figure 2.5 (a and b)). Commune used a digital stylus system for drawing and gesturing using a cursor, whereas VideoDraw used video links to enable whole-hand gesturing. In an evaluation of Commune using a two-person design task, the authors observed three novel aspects of the system: both collaborators shared the same orientation of the horizontal workspace; both collaborators were able to mark and point to exactly the same place in the workspace without interference from each other's hands; and collaborators were able to switch seamlessly between writing, drawing and gesturing. They also observed that cursors convey only pointing gestures rather than the rich variety of gestures observed in co-located collaboration, and that the drawing space provided by a monitor was too small.

VideoWhiteboard (Tang, J. C. and Minneman, 1991a) linked large vertical projected displays to support remote sketching. As with VideoDraw, collaborators could sketch simultaneously and see instant visual updates as others sketched. Instead of whole-hand gestures, collaborators now saw the shadows of their remote collaborators' entire bodies. The authors observed that the larger display now allowed collaborators to work side by side at each site. Although the shadows conveyed many of the gestures used in whiteboard interaction, they observed problems when pointing to precise locations, when conveying subtleties such as head-nods from a distance, and when collaborators' shadows overlapped. This may have been exacerbated by the lack of local feedback of the shadows presented to remote collaborators.

Ishii and Kobayashi (1992) later demonstrated ClearBoard, which provided the remote sketching task space of VideoDraw, combined with a head-and-shoulders video view of the remote collaborator presented on the same surface (Figure 2.5(c)). The design imitated collaboration through a clear glass board on which collaborators could draw. The camera at each site was positioned behind the surface in order to achieve greater spatial fidelity.

This work has continued more recently with commercial large-format remote sketching systems (e.g. SmartBoard, `http://smarttech.com/`) and investigation of shadow communication (Miwa and Ishibiki, 2004).

2.7.3 Remote Augmented Tangible Interaction Surfaces

Following early remote sketching systems, researchers created various large-format remote collaboration systems that augmented tangible task artefacts with visual information. Double DigitalDesk (Wellner and Freeman, 1993; Wellner, 1993a,b; Freeman, 1994) was one of the earliest such systems, following from the DigitalDesk project (Section 2.1.3). Each collaborator sat at their own desk and interacted with their own paper copy of the same page of information. The system augmented this paper with a video image showing annotations made by the remote collaborator, and an image of the remote collaborator's hands (Figure 2.5(d)). It followed TeamWorkStation (Ishii, 1990), which provided similar functionality but displayed the shared workspace on a conventional vertical monitor, rather than augmenting the paper on the desk itself. Tele-Graffiti (Takao et al., 2003), LivePaper (Robinson and Robertson, 2001), Agora (Kuzuoka et al., 1999), and PlayTogether (Wilson and Robbins, 2006) (Figure 2.5(e)) later pursued similar approaches. Of these systems, only Agora was evaluated with a user study. Luff et al. (2006) conducted an observational study of the system that focuses on the ways in which such systems can best support pointing gestures and referencing of tangible task artefacts.

Kirk (2007) also demonstrated a similar system for remote physical assembly tasks on horizontal surfaces. Designers' Outpost (Everitt et al., 2003) augmented sticky notes on a wall with projected imagery to support group-to-group remote collaboration in website design. Each group stood in front of their own wall, and both walls contained the same notes in the same positions. When sticky notes were added, removed or moved at one site, the same spot at the remote site was augmented with red light to highlight the inconsistency and indicate that the remote group should perform the same operation on their board. No detailed evaluation was conducted.

This problem of maintaining consistency between tangible task artefacts in workspaces at different sites affects all these systems, and is difficult to resolve. It is not clear how Double DigitalDesk, for instance, addresses the problem of one collaborator moving their copy of a shared document in the workspace, or turning the page of a multi-page document. Some interfaces, such as PlayTogether, have addressed this by opting not to replicate tangible task artefacts at each site. Instead, they have presented each collaborator with a projected image of tangible task artefacts that are not physically present at their own site. In such systems, only the collaborator who is co-located with a given task artefact can manipulate it; the other collaborators can only see it (typically at a low display resolution) and gesture to it. This asymmetry is inherent in tasks such as remote surgery, and advantageous in tasks such as remote bomb disposal. It may, however, be problematic for the kinds of tasks traditionally performed on tables, in which collaborators share task artefacts by passing them among themselves.

Brave et al. (1998) presented a novel technique for avoiding such inconsistencies and asymmetries between tangible user interfaces for remote collaboration. Each tangible task artefact was physically replicated at each site and, when a collaborator moved their local copy, the system used actuators to effect the same change for each of the remote copies. They have demonstrated two systems using such techniques: one moving blocks on horizontal surfaces using motorised chess boards; and another using rollers connected using force-feedback technology. Reznik and Canny (2001) demonstrated another implementation using vibrations and friction to move tangible task artefacts objects on a horizontal surface. At present, however, these technologies are limited to moving blocks and rollers, and it is not clear how they might extend to other interactions such as storage containers, adding and removing task artefacts, or sketching.

2.7.4 Remote Tabletop Interfaces

Just as early tangible interaction surfaces for co-located collaboration have been followed by tabletop interfaces (Section 2.1.4), the same trend has occurred in remote collaboration interfaces. Early remote augmented tangible interaction surfaces have been followed by large horizontal interactive surfaces in which all of the task artefacts are entirely non-physical and composed solely of computer-generated imagery. Two such surfaces are linked together to provide a shared workspace for remote collaboration. I refer to such systems as *remote tabletop interfaces*. Like tabletop interfaces, collaborators use direct input devices and are typically able to interact concurrently. Researchers have recently demonstrated a number of these systems, shown in Figure 1.2 (page 22).

Escritoire (Ashdown, 2004; Ashdown and Robinson, 2005) was one of the earliest such remote tabletop interfaces, and was partly inspired by the Double DigitalDesk work reviewed in the previous section. Instead of real paper, it used projected light to create "virtual sheets of paper" in a large horizontal shared workspace for remote collaboration, and so established the technical feasibility of remote tabletop interfaces. Two projectors were arranged to create a large (A0 size) low-resolution peripheral area, for storing virtual sheets, surrounding a smaller high-resolution foveal area, into which virtual sheets could be dragged for manipulation or reading. This arrangement was partly a product of the available display technology, and was inspired by the use of space on desks in offices rather than collaborative work practices. Remote collaborators used styluses and bimanual techniques to move and annotate "virtual sheets of paper", and to gesture to each other with telepointer traces. Three pairs trialled the system in a laboratory setting and completed questionnaires. They all agreed strongly that the shared workspace was useful, and were less sure that a video view showing the remote collaborator's face was also useful.

Though Escritoire established the technical feasibility of remote tabletop interfaces, it was inspired principally by the use of paper on single-person desks. The rationale for such a design in a collaborative setting seems unclear, and Escritoire was neither informed by, nor investigated, collaborative work practices. The single-fovea design, and a restriction that task artefacts cannot be rotated, limits the scope to sit collaborators around the table. Finally, there was little investigation of the feasibility of using the approach to address tasks beyond organising and annotating single "virtual sheets of paper".

RemoteDT (Esenther and Ryall, 2006) presented a shared workspace containing a Windows XP desktop. Remote collaborators used touch input to interact with legacy applications, to sketch, and to gesture to each other with telepointers. Multiple co-located collaborators could interact concurrently. TIDL (Hutterer et al.,

2006) provided similar functionality for legacy Java applications using multiple mice. Regenbrecht et al. (2006) demonstrated a system in which remote collaborators could move and rotate photos using touch input. They did not provide a remote gesture representation such as a telepointer or arm shadow, and the system did not allow multiple co-located collaborators to interact concurrently.

In VideoArms (Tang, A., et al., 2006a), remote collaborators could use touch input to move task artefacts, such as photos, and to sketch. Each collaborator's arms were captured using a camera and presented to remote collaborators as an image overlaid on the workspace. The system supported group-to-group collaboration since each surface was large enough for multiple co-located collaborators to stand around, though the touch input system did not allow co-located collaborators to interact concurrently. Digitable (Coldefy and dit Picard, 2007) provided similar functionality and used a multi-touch surface to remedy this problem. Lastly, C-Slate (Izadi et al., 2007) provided a shared workspace in which remote collaborators could use touch input to reposition task artefacts, such as virtual sheets of paper, and could annotate using a stylus. Like VideoArms and Digitable, each collaborator's arms were captured using a camera and presented on the surface to remote collaborators. The image of the arm became translucent as the arm was lifted from the surface. The surface was not sufficiently large for co-located collaborators to work side-by-side.

Although the systems are broadly similar, they differ in various ways: whether they focus on legacy applications or on the tabletop interaction techniques described in Section 2.2; the provision of remote gesture representations; and support for group-to-group collaboration. The systems also differ in their underlying motivation. Escritoire, for instance, was motivated by the affordances of paper on a single user's desk, and the opportunities to exploit large displays and bimanual input in remote collaboration technology. TIDL and RemoteDT were motivated by the potential to support group-to-group collaboration. However, their focus on legacy applications precludes the orientation techniques that have enabled successful around-the-table co-located tabletop collaboration. C-Slate and Digitable were motivated by the work practices exhibited in co-located tabletop collaboration (Section 2.3). However, C-Slate provides a much smaller form factor than typical co-located tabletop interfaces, and collaborators do not sit around it.

Many of the remote tabletop projects have focused on the exploratory construction of the technology itself. This is not insignificant given the considerable technical challenges involved. However, in spite of the technical progress, the approach has limited theoretical basis beyond allusions to the benefits of co-located tabletop collaboration and motivations of support for group-to-group collaboration. There has been little discussion of why a tabletop approach for remote collaboration might afford the work practices observed in co-located tabletop collaboration.

There are also few empirical studies of remote tabletop interfaces in use. Hauber et al. (2006) investigated a decision-making task and compared three conditions: co-located tabletop collaboration; a conventional remote collaboration interface with a shared workspace and a head-and-shoulders view of the remote collaborator; and remote tabletop collaboration, sat at opposite ends of the table, with a vertical screen showing the head-and-shoulders view of the remote participant. Participants in the remote tabletop condition talked more about the technology, took longer to complete the task, and reported being more confused than in the other conditions. However, in the remote tabletop condition they reported being more aware of where their remote partner was looking than in the conventional condition, and reported feeling more like they were in the same room. Pauchet et al. (2007) used Digitable to compare remote and co-located tabletop collaboration using a puzzle task. Remote collaborators again were provided with head-and-shoulders video of each other using vertical screens. They reported faster task completion times in the remote interface, though it is not clear why this effect arises, whether it is beneficial, or how it generalises to other tasks. Lastly, Tang, A., et al. (2006a) used the concept of workspace awareness to establish several hypothesised benefits of VideoArms over conventional telepointers. They conducted a brief exploratory study comparing the two representations using a tabletop interface connected to a remote wall display. The findings relate to remote arm representations rather than to remote tabletop interfaces per se, and are described in the following chapter.

Remote tabletop interfaces can also be compared to the other reviewed remote task-space technologies. They potentially support a wider variety of tasks than large-format remote sketching systems. Compared to remote augmented tangible interaction surfaces, remote tabletop interfaces do not permit tangible interaction with the task artefacts, and the view of the task artefacts is limited by the display resolution. However, they do not suffer the asymmetries and the consistency problem of the tangible systems.

In summary, researchers have recently demonstrated several remote tabletop interfaces. However, the projects differ in their motivation and design, and there are in some cases apparent inconsistencies between motivation and design. The reasons why the tabletop approach may be appropriate for remote collaboration, and the particular elements of co-located tabletop interaction that are important in a remote setting, remain without a theoretical basis or empirical evidence.

2.8 Theories of Remote Collaboration in Task Space

This section reviews theories of remote gesturing and awareness that I use as a theoretical basis for the design of remote shared workspace systems. I consider firstly language and remote gesture, followed by workspace awareness.

2.8.1 Language and Remote Gesture

A series of studies conducted at CMU investigated the benefits of shared workspaces for remote collaborative tasks (Kraut et al., 2002; Gergle et al., 2004, 2006). The studies used a two-dimensional instructor-follower puzzle task running in a conventional graphical user interface environment. Both collaborators saw a shared workspace showing the state of the task, but only the follower could interact with it, and only the instructor could see the target solution to be created. The collaborators were not able to gesture to each other. The first study empirically compared different system configurations, including configurations in which the system presented no shared workspace, forcing collaborators to rely solely on audio communication (Kraut et al., 2002). The shared workspace improved performance, and this improvement was degraded if shared workspace was subject to a 3s delay. The improvement was more marked when the task space was visually complex (such as puzzle pieces overlapping in the target puzzle, or puzzle pieces changing colour through the task). They later explained these results using a detailed analysis of language use in a similar study (Gergle et al., 2004). They showed that the shared workspace enabled visual actions to replace parts of the dialogue that would otherwise have been necessary. In particular, the instructor used the actions of the follower in the shared workspace to infer whether instructions had been understood and carried out correctly. This otherwise required explicit questioning and back-channel responses. The final experiments examine the effects of delayed feedback in the shared visual workspace (Gergle et al., 2006).

Kirk (2007) criticises this work as “devoid of significant implication for the actual deployment of technologies”. He argues firstly that its reductionist approach has led to artificial tasks bearing little resemblance to real-world instructor-follower tasks, which tend to be three-dimensional, or to other real-world collaborative activities such as design tasks. Furthermore, he argues that many of the findings are “somewhat expected” and in any case assumed by technology designers. In particular Clark’s theory of conversational grounding (Clark and Brennan, 1991) considered how delay and the costs of referring to task artefacts and repairing misunderstandings affects efficient language use in different communication media.

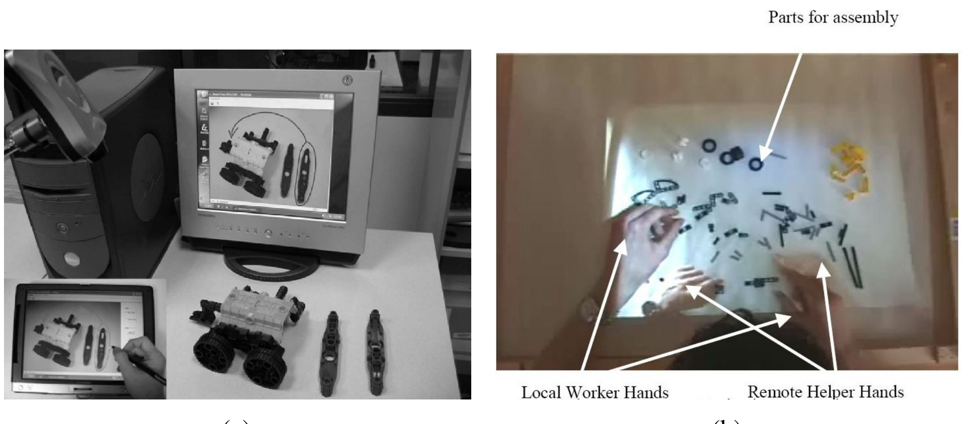


Figure 2.6: Examples of remote gesture technologies: (a) using sketching (from Fussell et al., 2004); and (b) using an unmediated video representation of the hands (from Kirk, 2007).

The above work investigated shared workspaces in the absence of remote gesturing. A number of studies have investigated the generation and presentation of remote gestures in shared workspace systems, and the effects on language use. Kirk (2007) provides a comprehensive review. Fussell et al. (2004) investigated a remote gesture system called DOVE, which presented on a monitor digital sketches superimposed on a video view of the physical workspace (Figure 2.6(a)). Using a remote physical assembly task, they found that DOVE provided performance benefits over a video-only condition with no gesturing, while a basic cursor pointing system did not. They attribute this to the support in DOVE for more complex gestures such as to illustrate actions that would otherwise require lengthy referential utterances. Kirk et al. (2007) found that an unmediated video representation of hands (Figure 2.6(b)) also yielded performance benefits over a video-only condition with no gesturing. Their analysis of language showed that gesturing enabled deictic utterances that may have replaced lengthier descriptive utterances, and that gesturing enabled turn-taking. In a further study, Kirk and Fraser (2006) found in a remote physical assembly task that an unmediated video representation of the hands yielded faster task completion than a sketching approach or a combination hands and sketch approach, with no loss of accuracy. They argued that sketching creates an unnecessary level of abstraction, and show in other work that the hands-only condition can convey a variety of complex gestures (Kirk et al., 2005).

Further work demonstrates that successful remote gesturing requires more than simply a gesture surrogate in the workspace. In an environment with various task artefacts, the gesturer must be sure that their gesture both attracts the attention of their collaborator and prompts their collaborator to look at the appropriate task artefact or part of a task artefact. This process is known as securing a common alignment or orientation. Gaver et al. (1993) report the difficulties experienced by participants in securing alignment in a system that allowed collaborators to choose between multiple disjoint camera views. In particular, it was difficult to attract attention because the collaborator might be looking at a different camera view that did not show the gesture. Heath et al. (2001) studied Gestureman, a robot gesture surrogate with a laser pointer. They found that collaborators found it difficult to secure alignment because of the "always on" nature of the laser pointer, and also because collaborators could not infer each other's fields of view. Luff et al. (2006) investigated a system with unmediated hand gestures and observed that seeing the trajectory of a gesture as it unfolds enables the remote collaborator anticipate its time and position of arrival.

2.8.2 Workspace Awareness

The ability to maintain an awareness of collaborators' actions in the workspace is central to many collaborative tasks, whether working loosely or tightly coupled. Co-located collaborators use a variety of visual and auditory cues to maintain awareness, whereas remote collaborators are reliant on the cues that must both arise through interaction with the system and also be conveyed by the system. Gutwin and Greenberg (2002) describe how "[poorly-designed remote collaborative systems] often seem inefficient and clumsy compared with face-to-face work". Consequently, the generation, presentation and use of awareness cues are important factors in the design of a remote collaboration system. The authors present a theory describing the information comprising workspace awareness, the perceptual cues that give rise to this information, and how collaborators use the information in practice. The theory is grounded in a variety of sources: observations from investigations of workspace awareness in conventional groupware systems (e.g. Dourish and Bellotti, 1992; Greenberg et al., 1996; Gutwin et al., 1996; Gutwin and Greenberg, 1999); observational studies of co-located collaboration such as tabletop design tasks, control rooms and aircraft flight decks (e.g. Heath and Luff, 1992; Segal, 1994); and theories of collaboration outside conventional CSCW. This section provides an overview and examples.

Gutwin and Greenberg (2002) define *workspace awareness* as "the up-to-the-moment understanding of another person's interaction with the shared workspace", building on definitions of situation awareness used in human factors research. They describe the information comprising workspace awareness at any point in time as follows: who is working in the workspace, their identity and attribution of their actions; their current action, intention and task artefact under manipulation; the location of their work and gaze; their field of view; and their ability to reach. This is augmented with information relating to prior actions. They demonstrate that this framework can inform the design of conventional groupware systems, such as the provision of miniature workspace overviews indicating the position of each collaborator's current viewport and telepointer (Gutwin and Greenberg, 1999).

They then describe three sources of workspace awareness cues:

- *Consequential communication* arises from the movement of a collaborator's body as a consequence of their actions in the workspace. These cues are perceived as collaborators watch each other work, either peripherally or intentionally.
- *Feedthrough* describes the visual and auditory cues arising from manipulation of a task artefact. Just as feedback informs the manipulator, feedthrough informs their collaborators.
- Lastly, the theory considers *intentional communication* such as conversation and gesture.

Consider as an example the movement of a coffee cup: consequential communication is provided by the movement of the arm towards the cup with the outstretched fingers to grab the handle, and the subsequent movement of the arm; and feedthrough is provided by the movement of the cup itself and the noise as it is placed on another surface.

They also describe how collaborators use workspace awareness information in collaboration. It enables collaborators to manage the transitions between different collaborative coupling styles. Collaborators can, for example, work loosely coupled while maintaining an awareness of each other's actions and later to transition to tight coupling when they become aware that such a transition could be beneficial. Workspace awareness also offers opportunities to simplify verbal communication that would otherwise be inefficient, through the mechanisms described in the previous section. Collaborators also use workspace awareness for implicit coordination, the process of ensuring that actions occur correctly with respect to each other and to the task constraints. This process takes place both at the fine-grained level, such as interleaving manipulations of task artefacts by different collaborators, and also at the coarse level, such as planning and reorganising the division of labour in an activity. Lastly, workspace awareness enables collaborators to anticipate each other's actions and also to assist each other. This framework has been applied to derive experimental measures of awareness that have been used to investigate the effects on awareness of various interventions in a conventional groupware system (Gutwin and Greenberg, 1999).

In a separate paper, Gutwin and Greenberg (1998) have argued that interventions to increase workspace awareness in conventional groupware systems typically reduce the power of individual collaborators. They consider three such interventions: adding workspace overviews showing the actions and fields of view of other collaborators; making actions more perceivable; and constraining all users to a single view of the underlying workspace data. However, they argued, the overview reduces the screen space available for the detailed viewport into the workspace that is used in individual work; perceivable actions preclude powerful individual interaction techniques such as keyboard shortcuts; and the single view constraint limits individual users when they would otherwise switch to different views.

2.9 Discussion and Chapter Summary

In this chapter, I have reviewed the prior work in order both to identify the research opportunities and to provide a context for the work that follows. I conclude by summarising the two main areas of consideration.

Firstly, tabletop interfaces for co-located collaboration are large horizontal interactive surfaces that present collaborators with computer-generated imagery with which they can interact, typically concurrently, using direct input devices. They arose from prior work in single display groupware and other large-format direct input displays, and from studies of collaboration at conventional tables. Their interaction design is motivated by analogies to paper task artefacts on a conventional table, and the need to support multiple collaborators around the table. They tend to present small task artefacts that can be moved, reoriented and regrouped. Explicit access controls for mediating individual and group work have not been necessary in a variety of applications. Co-located collaborators at tabletop interfaces can exhibit work practices observed in collaboration at conventional tables, such as carrying out both individual and group work, and using spatial partitioning and orientation to mediate interactions. However, the range of applications demonstrated so far has been rather limited.

Secondly, remote tabletop interfaces are linked tabletop interfaces that provide a shared workspace for remote collaboration between individuals or groups. They arose from prior work in large-format remote sketching interfaces and remote augmented tangible interaction surfaces that were, in turn, motivated by studies of collaboration at conventional tables. However, the limited systems that have been demonstrated have varied in the motivation, interaction design and provision of remote gesture representations. In contrast to the conceptual frameworks and experimental and observational studies of co-located tabletop collaboration, and shared workspaces more generally, there is a lack of sound reasoning and evidence for why the tabletop approach is appropriate in a remote setting, and how such interfaces should be designed. Again, the range of applications demonstrated so far has been limited.

Chapter 3

Design of Remote Tabletop Interfaces

In the previous chapter I reviewed work on remote tabletop interfaces: large horizontal displays that provide shared workspaces for synchronous remote collaboration between individuals or groups, and in which collaborators can interact concurrently. They are distinct in physical form and technical construction from other shared-workspace remote collaboration interfaces such as whiteboards, remote tangible interfaces and conventional groupware systems. Beyond this, however, there has been little discussion or evaluation of remote tabletop interfaces, or of how and why they differ in practice from conventional groupware systems. There is little theoretical basis for adopting a tabletop approach in a remote setting, and there are few design guidelines for doing so.

In this chapter, I apply theories of awareness and work practices to remote tabletop interfaces in order to develop design guidelines and questions that will be investigated empirically in Chapter 6. I then discuss the technical implications of these guidelines in order to inform the technical work in the following chapter.

3.1 Designing for Workspace Awareness

Gutwin and Greenberg (2002) described how users of conventional groupware systems experience problems maintaining awareness of each other's actions. Their framework, reviewed in Section 2.8.2, defines workspace awareness; discusses how collaborators use it; and describes the sources of workspace awareness as consequential communication (movement of a collaborator's body as they interact), feedthrough (visual changes in task artefacts in the workspace as a collaborator interacts), and intentional communication. In this section, I apply this awareness framework to consider the design of remote tabletop interfaces, discussing in turn remote representations of bodies, interaction with task artefacts, and workspace navigation.

3.1.1 Remote Representations of Bodies

Previous remote tabletop interfaces have employed a variety of remote body representations: high-fidelity arm images or “shadows”, telepointers, and telepointer traces. Their most obvious role is to convey communicative and workspace gestures, whose use in design tasks were reviewed in the previous chapter. Tang, A., et al. (2006a) hypothesised that such gestures are best conveyed by a representation that captures the fine-grained movements of the hands and is displayed in the workspace itself. Impoverished representations such as telepointers and telepointer traces, they hypothesised, do not allow easily convey communicative gestures, nor workspace gestures other than pointing. This is supported by their observations of gesture in a study comparing high-fidelity arm images and telepointers. Kirk et al. (2005) also observed effective remote gesturing in a remote physical assembly task using high-fidelity arm images displayed in the workspace. This leads to the first hypothesised awareness mechanism:

Q1a. High-fidelity representations of arms, displayed in the workspace, enable rich gesturing for intentional communication.

and the first design guideline:

G1. Display high-fidelity arm representations in the workspace.

Tang, A., et al. (2006a) applied the awareness framework of Gutwin and Greenberg (2002), and hypothesised that direct input devices, combined with high-fidelity remote arm representations in the shared workspace, additionally enable effective consequential communication. Collaborators are able to visualise each other’s atomic level interactions with the workspace in order to infer the actions being generated, and can see how each other’s devices are being manipulated. Impoverished, indirectly-controlled representations such as telepointers do not, for instance, allow collaborators to see each other reaching with their hands for a task artefact or tool, or retracting their arms from the display to think. They observed that remote collaborators using remote arm representations and direct input devices watched each other work for considerable periods of time while performing design and puzzle tasks, which supports this hypothesis. This leads to a fourth design guideline:

G2. Use direct input devices such as styluses or direct touch.

I propose, however, that these first four design guidelines alone (large horizontal displays, concurrent interaction, remote arms, and direct input) are insufficient to convey effectively consequential communication. I propose that effective consequential communication depends additionally on the way in which the interface enables collaborators to interact with task artefacts in the workspace.

3.1.2 Interaction with Task Artefacts

I propose that two further guidelines are necessary to constrain the interaction design in order to convey effective convey consequential communication. Firstly, Gutwin and Greenberg (1998) proposed that interaction in conventional groupware systems should follow the direct manipulation principles of Shneiderman (1983) so that actions result in immediate incremental, continuous visual changes in the workspace. I propose the same should hold for remote tabletop interaction. Secondly, I propose that remote tabletop interfaces should use interaction techniques that localise the visual effects of an action in the shared workspace to the vicinity of the interacting hand. I summarise these two further guidelines as follows:

G3. Interaction should follow direct manipulation principles to result in immediate incremental, continuous visual changes, such as dragging of task artefacts.

G4. Interaction should be localised to the vicinity of the hand, such as dragging of small task artefacts.

In order to illustrate these two further guidelines, consider the dragging of a small virtual task artefact. The location and velocity of the task artefact are coupled to those of the arm causing the action (and its remote representation), and so remote collaborators can attribute the dragging motion of the task artefact to the person causing it as the action unfolds. The localisation also enables collaborators to anticipate the task artefacts that their partners might drag, based on their arm locations and reaching motions. Finally, the incremental changes in the appearance of the workspace can be observed as the action unfolds, rather than afterwards, enabling effective feedthrough.

Compare this to, for example, a remote tabletop application that presents a map in which an extra layer of data can be made instantly visible across the entire map by changing the hand into a fist gesture and tapping anywhere on the map. The visual effects of the action are not localised to the area of the hand, and so it is harder to determine who has caused such an action. The action can be instigated anywhere and so it cannot be anticipated based on arm locations and reaching actions. Both the touching action (consequential communication) and the display change (feedthrough) are near-instantaneous and are unlikely to be noticed by the collaborators until the action is complete. Only the clenching of the fist is therefore perceivable as the action unfolds.

Consider instead an alternative technique in which the new data layer must be dragged down from one side of the table like a window-blind. The action is more localised and incremental than the fist-tapping example. It can be anticipated as the instigator reaches across the table to the appropriate side, and can be perceived as it unfolds in the dragging movement of the arm coupled with the movement of the data layer.

Accordingly, I summarise the two awareness mechanisms as follows:

Q1b. Direct manipulation techniques enable task artefact feedthrough.

Q1c. High-fidelity representations of arms, displayed in the workspace, together with direct input devices, enable consequential communication when used with local, direct manipulation techniques.

Although the awareness value of these two additional design guidelines (G3 and G4) in remote tabletop systems has not been postulated, they have often been fulfilled in practice in large collaborative display systems. In collaboration around conventional tables, for example with paper photographs, the direct manipulation is dictated by physics, and the localisation by their relatively small size. Similarly, the guidelines also apply to the interactive tabletop systems for digital photos that aim to mimic the physical world (e.g. Shen et al., 2002; Scott et al., 2005; Apted et al., 2006). Both traditional whiteboards and digital whiteboards localise the ink to the pen, and the marks on the board change incrementally.

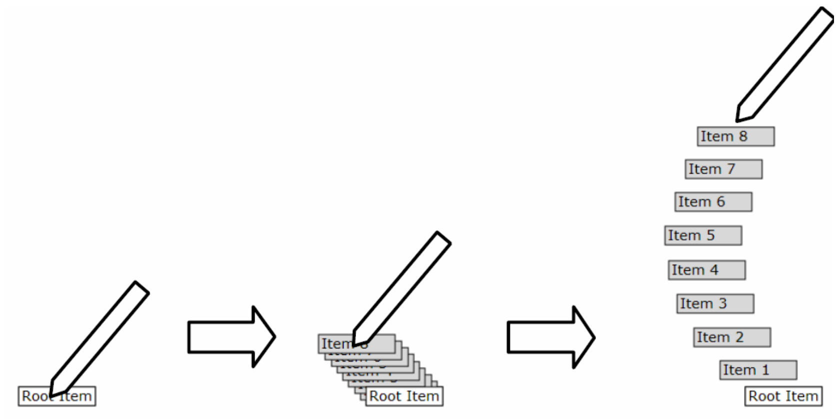
Large collaborative display interaction is not always possible using only sketching or dragging of small task artefacts. However, like the window-blind example, a number of other tabletop interaction techniques follow the two proposed guidelines of localised, direct manipulation:

- *Task artefact creation*. In a conventional groupware system, task artefacts are typically created by clicking in a fixed location or on a menu. Consider, for example, clicking on a file icon to open it, or clicking on a link to open a web page in a new window, or using a menu to insert a graphic or new page into a wordprocessing document. These actions are symbolic rather than direct manipulation. Task artefacts could instead be created by dragging from a pile or storage container (e.g. Ashdown and Robinson, 2005; Hinrichs et al., 2005; Scott et al., 2005). Where this is not appropriate, task artefacts could instead by created by dragging outwards from a particular point (Figure 3.1(a)) (Leithinger and Haller, 2007).

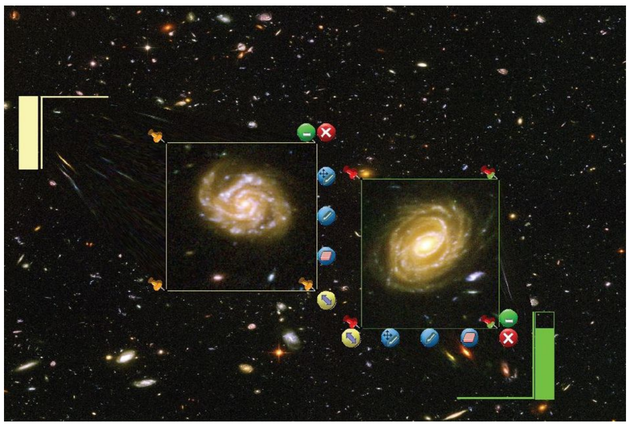
- *Menu opening*. Clicking to open a menu is similarly symbolic, and also risks obscuring another collaborator's work. Tabletop interfaces have instead demonstrated permanently-open moveable tool palettes as an alternative to non-context-sensitive menus (Morris et al., 2006b). A *Toolglass* is similarly an alternative to a context-sensitive menu. It is a translucent or transparent always-open task artefact that can be dragged into position over the top of the intended point of action in the workspace, before instigating the action by "clicking through" it (Figure 3.1(c)) (Bier et al., 1993). Such menus could also be opened by dragging outwards from a point (Leithinger and Haller, 2007). (This technique differs from radial menus, also known as pie menus or marking menus, in which menu items are selected by dragging outwards from a point (Wiseman et al., 1969)).

- *Mode selection for visualisation*. Selecting a visualisation mode in a conventional shared workspace system typically involves selecting from a list, which then causes the entire workspace to change. Consider, for example, a map that allows different layers to be made visible, or switching between printing and editing views in a word processor. Such actions are neither direct manipulation nor local. An alternative approach uses smaller movable visualisation lenses (Bier et al., 1993) for each collaborator, within which the visualisation mode can be changed without affecting the rest of the workspace 3.1(b) (Forlines and Shen, 2005; Tang, A., et al., 2006b). The window-blind technique also addresses this problem.

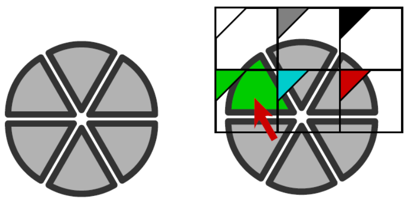
- *Task artefact removal*. In conventional groupware systems, task artefacts are typically removed by symbolic techniques such as clicking or using a keyboard shortcut. Apted et al. (2006) demonstrated a system in which task artefacts are instead moved to a "black hole" or "recycle bin", which may be more appropriate on a large collaborative display.



(a)



(b)



(c)

Figure 3.1: Examples of direct manipulation interaction techniques that localise the action to the vicinity of the instigating hand or pointer: (a) creating task artefacts by dragging from a point (from Leithinger and Haller, 2007); (b) using multiple lenses (from Forlines and Shen, 2005); and (c) toolglasses (from Bier et al., 1993).

There are also situations in which the additional guidelines cannot be applied. Dragging a large task artefact, for example, is direct manipulation but cannot be localised to the vicinity of the interacting hand because of the large size. Similarly, the lens or window-blind techniques may not always be appropriate, and instead a single action to change the visualisation mode for the entire display may be necessary. If the visual effects of an action cannot sensibly be localised to the vicinity of the interacting hand, then awareness may instead be promoted by localising the area in which the action can be instigated. Collaborators can then see each other reaching towards this area and anticipate the action. Consider, for instance, using a small "drag handle" attached to the large task artefact, or a small mode selection control in a fixed location on the display. Morris et al. (2006b) used a similar argument when considering a centralised menu in tabletop collaboration, versus replicated menus for each collaborator.

Applying these guidelines in practice may boost awareness but negatively impact other areas, presenting a trade-off. They may, for example, rule out expressive, novel interaction techniques. Gutwin and Greenberg (1998) observed that replacing symbolic actions, such as keyboard shortcuts, with more perceivable direct manipulation benefits group awareness but increases the effort required by individuals. They argue that this would be unacceptable to users and that consequently symbolic actions cannot be entirely removed from conventional groupware systems. They instead suggest mechanisms for more perceivable symbolic actions without direct manipulation, such as animations. Localising interaction may present a similar trade-off between awareness and individual effort: although it encourages reaching, which may boost awareness through consequential communication, this extra reaching likely requires more effort by individuals. If the controls being reached for cannot be moved close to the user then frequent repetitive reaching may, over time, become fatiguing. (Though Morris et al. (2006b) conducted a study of centralised controls on a tabletop interface and reported that no participants complained about reach ergonomics). Some of the potential ergonomic strain of reaching may be reduced by using an input technology that allows users to lean on the table with their elbow while they reach, as they can on conventional tables.

Direct manipulation, localised to the vicinity of the interacting hand, is intended as starting point for considering awareness in remote tabletop interaction techniques. I anticipate further investigation of this area as different tasks are considered and more interaction techniques are developed. Consider, for example, an interface in which a collaborator is able to manipulate a task artefact in different ways depending on the manipulation mode they selected from a menu beforehand. An application to support painting, for instance, may allow a user firstly to select from a floating tool palette to decide whether to draw a shape or to move a shape, and then secondly drag in the workspace to perform the manipulation. Such an interaction technique is both direct manipulation and localised. However, the shape and location of the arm in the second stage provide no cues as to which manipulation mode (drawing or moving) is selected. Alternative techniques may provide greater workspace awareness through consequential communication or feedthrough, such as mode selection using hand or stylus gestures, or by using an additional "tool" task artefact such as a toolglass as described above (Figure 3.1(c)) (Bier et al., 1993). Consider, as a more radical example, a tabletop interface to a text-based programming language. Each method or functional unit could be presented as a movable task artefact that collaborators can edit using individual keyboards. This scheme may provide awareness information about the methods or functional units that each collaborator is currently using. However, collaborative programming is very different to the tasks previously investigated using tabletop interfaces, and so it is difficult to reason about whether such an interface would be effective in practice. Accordingly, localised direct manipulation is intended only a starting point for investigation, and further development will be necessary as different tasks are considered.

3.1.3 Workspace Navigation

Conventional groupware systems, and conventional applications more generally, typically use the screen as a scrollable viewport into a much larger workspace containing rigidly-arranged, automatically-arranged task artefacts. Consider a typical word processor, photo sorting, or spreadsheet application. Early conventional groupware systems used a strict "What You See Is What I See" (WYSIWIS) approach, in which all collaborators' screens show the same region of the shared workspace at any given time (Stefik et al., 1987b). This prevents collaborators from navigating independently to work in different areas of the workspace. Later systems consequently relaxed the WYSIWIS constraint to allow independent workspace navigation by scrolling (Figure 3.2) (e.g. Gutwin and Greenberg, 1998) or minimising areas (Stefik et al., 1987a). However, such systems provide poor awareness of actions occurring outside each collaborator's own viewport, which consequently impairs their ability to switch between independent and group working (Gutwin and Greenberg, 2002). This is particularly detrimental in *mixed-focus tasks*, which require frequent switching. A miniature overview of the entire workspace, indicating the other collaborator's pointer and viewport, helps remedy this problem and is currently the state of the art (Figure 3.3) (Gutwin and Greenberg, 1999). The size of the overview determines whether the design favours group or independent work (Gutwin and Greenberg, 1998).

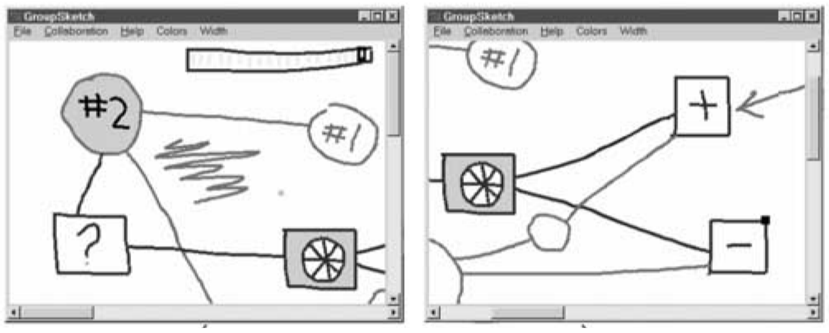


Figure 3.2: Independent scrolling using a relaxed WYSIWIS approach to workspace navigation (from Gutwin and Greenberg, 2002).

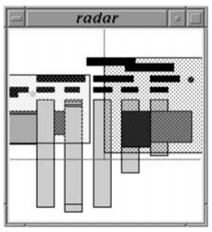


Figure 3.3: Miniature overview that aids workspace awareness (from Gutwin and Greenberg, 2002).

Many co-located and remote tabletop interfaces adopt a different approach. They restrict the size of the workspace to the physical size of the display, eliminating scrolling. This is possible through two strategies, shown in Figure 3.4. Firstly, tabletop interfaces are physically large compared to conventional screens. This allows a larger visible workspace and side-by-side working by multiple collaborators. Secondly, in mimicking interactions with real-world task artefacts, tabletop interfaces tend to allow collaborators to arbitrarily position and overlap task artefacts to create piles (e.g. Ashdown, 2004). Accordingly, more task artefacts can be presented in a given size of workspace than the automatic, rigid arrangement in conventional systems. Similar approaches include visualisation techniques such as shrinking task artefacts at the far side of the table (Guimbretière et al., 2001; Shen et al., 2004) or in special storage containers (Scott et al., 2005; Apted et al., 2005). A rigidly-positioned viewport and movable task artefacts contrasts with the conventional approach of rigidly positioning the task artefacts and movable viewport.

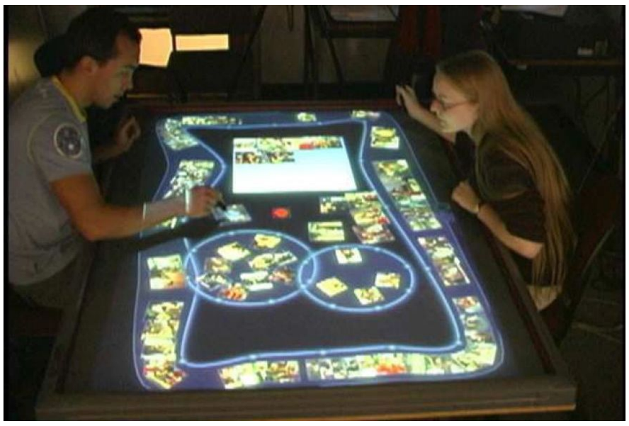


Figure 3.4: Example showing how tabletop interfaces can display many task artefacts yet avoid workspace navigation by exploiting their large size and tabletop interaction techniques such as shrinking task artefacts (from Hinrichs et al., 2005).

Together these two strategies allow many remote and co-located tabletop interfaces to avoid workspaces larger than the physical display size, and so avoid scrolling and the associated problems. I propose that employing these strategies in remote tabletop interfaces enables remote collaborators to carry out independent work in different parts of the workspace as part of the task, while nevertheless maintaining a high level of awareness of each other's actions in the workspace. At all times they can see each other's gestures, consequential communication, and feedthrough, regardless of where in the workspace they are working. Such a level of awareness is unlikely to be achievable using the conventional approach, even with a miniature overview, since its small size limits the fidelity and detail with which gestures, consequential communication, and feedthrough can be displayed.

Accordingly, I summarise the final hypothesised awareness mechanism and design guideline:

Q1d. Avoiding workspace navigation, by using large displays and appropriate interaction, enables collaborators to work independently in different areas of the workspace while maintaining workspace awareness.

G5. Avoid workspace navigation by using large displays and appropriate interaction design, such as casual piling and shrinking.

Not all tabletop applications use these strategies to eliminate workspace navigation. Collaborative applications that require large task artefacts, such as maps, or task artefacts that are necessarily spatially rigid with respect to each other, such as a data visualisation structure, are not able to use piling and shrinking mechanisms. Alternative data visualisation techniques may nevertheless reduce the size of the workspace in many cases. The map or structure might, for instance, be shrunk to the size of the display and explored using individual movable fish-eye lenses (Forlines and Shen, 2005). Some large task artefacts, such as a text document, may be split into smaller pages, such that only a single page is visible at any time. Alternatively, a larger table may be employed.

Applications requiring task artefacts that are too large and too detailed to use these approaches may necessitate larger displays or may not be able to avoid workspace navigation.

3.1.4 Summary

I have applied the awareness framework of Gutwin and Greenberg (2002) to remote tabletop interfaces to examine gesture, consequential communication and feedthrough in remote body representations, interaction design and navigation. This analysis has resulted in the following question and hypothesised awareness mechanisms:

Q1. Which of the following mechanisms contribute to workspace awareness?

- **Q1a.** High-fidelity representations of arms, displayed in the workspace, enable rich gesturing for intentional communication.
- **Q1b.** Direct manipulation techniques enable task artefact feedthrough.
- **Q1c.** High-fidelity representations of arms, displayed in the workspace, together with direct input devices, enable consequential communication when used with local, direct manipulation techniques.
- **Q1d.** Avoiding workspace navigation, by using large displays and appropriate interaction, enables collaborators to work independently in different areas of the workspace while maintaining workspace awareness.

These hypothesised awareness mechanisms, if true, lead to the following design guidelines:

G1. Display high-fidelity arm representations in the workspace.

G2. Use direct input devices such as styluses or direct touch.

G3. Interaction should follow direct manipulation principles to result in immediate incremental, continuous visual changes, such as dragging of task artefacts.

G4. Interaction should be localised to the vicinity of the hand, such as dragging of small task artefacts.

G5. Avoid workspace navigation by using large displays and appropriate interaction design, such as casual piling and shrinking.

3.2 Tabletop Work Practices

Having established design guidelines to promote workspace awareness, I now examine the work practices that have been observed in co-located tabletop interfaces, and consider whether each might apply in remote tabletop collaboration.

3.2.1 Collaborative Coupling

Section 2.8.2 described how collaborators use workspace awareness information to manage collaborative coupling in order to transition appropriately between working together and working independently on the same task (Gutwin and Greenberg, 2002). This behaviour has been observed in co-located collaborative design tasks at conventional tables, as described in Section 2.3.1 (Tang, J. C., 1991; Kruger et al., 2004; Scott et al., 2004). A remote tabletop system that provides a high level of workspace awareness might therefore enable effective management of collaborative coupling and accordingly support tasks, such as collaborative design, that involve these transitions.

This hypothesis is supported by experiences with tabletop interfaces for co-located collaboration that broadly follow the proposed guidelines. Section 2.3.1 described how two studies have observed support for a range of different coupling styles. Tang, A., et al. (2006b) observed pairs undertaking spatial data tasks using individual lenses. They were able to transition frequently and fluidly between a variety of coupling styles, ranging from tightly-coupled working together through to working completely independently. Similarly, Scott et al. (2005) observed pairs undertaking a photo sorting and arrangement task using movable interactive storage containers. The containers allowed the pairs to work both together and individually. Neither study reported difficulties with either maintaining awareness or transitioning opportunistically.

3.2.2 Spatial Partitioning

Section 2.3.2 described how co-located collaborators implicitly divide the physical space on the table surface. It reviewed two studies showing that this territorial behaviour serves coordination roles in collaboration (Tang, J. C., 1991; Scott et al., 2004). It enables each collaborator to reserve an area of the table for individual work. The distinction between personal and group territory establishes an audience for actions, and mediates transitions between coupling styles. Task artefacts can be reserved and handed back to the group by moving them between personal and group territory. Lastly, collaborators implicitly divide group areas to take responsibility for the nearest region.

The findings of Scott et al. (2004) suggest that territoriality in co-located tabletop collaboration relies on visibility of action, sufficient physical space, movable task artefacts and direct input mechanisms. These factors are embodied in the design guidelines that I have proposed for remote tabletop interfaces, but it remains unclear whether they afford territorial behaviour. One important factor is seating arrangement.

3.2.3 Seating Arrangement

Remote collaborators, in order to be territorial, would need to sit in different places around the shared workspace so that each had space for a personal territory. This seating arrangement, combined with direct input devices, ensures that a collaborator has to physically reach across the table to access another collaborator's personal territory. Remote arm representations additionally make this reaching action visible to that collaborator and emphasise the seating arrangement.

However, remote tabletop collaborators can also arrange themselves so that they both sit in the same position relative to the workspace. Collaborators in this second, *overlaid*, arrangement are unlikely to behave territorially since their personal territories overlap, and they may consequently experience coordination difficulties. This hypothesis is supported by two social psychology studies suggesting that co-located collaborators prefer to sit at a distance that allows them to use territory to mediate their interactions (reviewed by Scott et al. (2004)). Hall (1966) explained that collaborators are most comfortable working at "arm's length" from others, a distance that preserves their personal space. In a study of a school library, Thompson (1973) reported that students preferred rectangular tables to round tables because it was easier to establish individual space.

Kirk (2007), however, studied seating arrangement in a remote physical system using an instructor-follower task and found trends, though not statistically significant, that an overlaid seating arrangement was easier for participants and resulted in more progress than other seating arrangements. However, personal territories were not advantageous because the task was neither open-ended nor required frequent switching between independent and group work, and therefore did not require such coordination mechanisms. Furthermore, personal territories were not possible in such a system because only one collaborator could manipulate task artefacts.

Accordingly, though I hypothesise that territoriality relies on a non-overlaid seating arrangement, it remains unclear which seating arrangement is preferred by collaborators in practice.

3.2.4 Task Artefact Orientation

Task artefact orientation, introduced in Section 2.3.3, is a third practice associated with co-located tabletop collaboration. The orientation of a task artefact on the table establishes its ownership and availability, helps establish territories, and can gain a collaborator's attention (Kruger et al., 2004). The Rotate'N'Translate interaction technique (Section 2.2.2) (Kruger et al., 2005) allows casual reorientation of task artefacts and is designed to enable collaborators to use these coordination mechanisms in interactive tabletop interfaces.

It is unclear whether task artefact orientation serves these roles in remote tabletop interfaces. Like territoriality, it would rely on remote collaborators being seated around the shared workspace so that a task artefact oriented to one collaborator is not oriented to any other. Hauber et al. (2006) briefly discussed a remote tabletop system in which collaborators position and orient digital photos as part of a wider study of presence. They found that collaborators oriented the photos for comprehension but, perhaps because of the closed nature of the task, not for coordination.

3.2.5 Summary

By considering work practices in co-located tabletop collaboration, I have identified the following additional questions about work practices in remote tabletop collaboration:

Q2. Does the hypothesised high level of awareness enable collaborators to work in a range of coupling styles, with frequent, fluid transitions between, similar to co-located tabletop collaboration?

Q3. Do remote tabletops support territorial partitioning of space as a coordination mechanism?

Q4. How is territorial partitioning affected by an overlaid or non-overlaid seating arrangement?

Q5. Do remote tabletops support task artefact orientation as a coordination mechanism?

Q6. How is task artefact orientation affected by an overlaid or non-overlaid seating arrangement?

3.3 Technical Requirements and Feasibility

In this section I discuss the technical requirements for a remote tabletop interface that meets the guidelines, and the technical challenges of constructing such a system. I also identify two further practical requirements for rapid exploration of useful remote tabletop interfaces: support for high display resolutions; and support for the reuse of existing user interface components.

3.3.1 Tabletop Interaction

Although creating a large horizontal display surface is relatively trivial using a projector, conventional graphical user interface systems and applications lack support for tabletop interaction. They do not, for instance, support concurrent interaction among multiple collaborators, provide software interfaces for direct input devices (G2), or provide programming abstractions to support local, direct manipulation with task artefacts that can be casually moved, shrunk, and grouped (G3, G4 and G5).

These requirements for tabletop interaction are currently addressed by tabletop infrastructure software such as DiamondSpin (Shen et al., 2004), the Buffer Framework (Isenberg et al., 2006) and Cruiser (Apted et al., 2006), as well as several other implementations that have been created to investigate interaction techniques such as Interface Currents (Hinrichs et al., 2005).

3.3.2 Remote Collaboration

These tabletop systems do not, however, provide the requisite support for remote collaboration, such as a shared workspace at different sites, or remote arm representations (G1). Nor can such functionality easily be added retrospectively. Consider, for example, augmenting such a system with a simple remote display protocol that captures each rendered frame and forwards the pixel data to another computer for remote collaboration. Linking displays in this naïve way at standard XGA resolution (1024px×768px) at 30fps using 32 bits per pixel requires network bandwidth of 0.75Gb/s. This is beyond the capacity of the network hardware and links that are practically available to user interface researchers, particularly in a field deployment connecting different sites. Remote display protocols such as VNC (Richardson et al., 1998) use compression techniques to avoid this problem, but these techniques are designed for conventional graphical user interface properties in which windows remain axis-aligned. Applying them to a tabletop system in which task artefacts appear rotated and scaled is likely to result in considerable performance degradation. Furthermore, such protocols do not effectively support multi-user input.

Accordingly, instead of retrospectively adding a remote display protocol, the requirements necessitate an architecture that is designed for remote collaboration and exploits abstractions to achieve performance over limited bandwidth. Remote collaboration systems such as TIDL (Hutterer et al., 2006), RemoteDT (Esenther and Ryall, 2006) and GroupKit (Roseman and Greenberg, 1996) are designed to support multi-user input between sites. However, they use conventional graphical user interfaces rather than tabletop interaction, and again, this functionality is difficult to add retrospectively.

3.3.3 Display Resolution

The range of demonstrated tabletop applications, reviewed in Section 2.1.4, has so far been rather limited. There has been little investigation of tabletop applications to support the collaborative tasks for which people currently use their desktop computers, such as collaborative web browsing, document review and data analysis. These are compelling applications to which tabletop interfaces may bring significant benefits, and this is especially true when considering remote collaboration. The photo-based tabletop applications that have been demonstrated so far for co-located collaboration are useful for investigating work practices and interaction techniques, and may ultimately become widespread for serendipitous interaction in museums, coffee shops and homes. For remote collaborators, however, applications like collaborative data gathering, text document annotation and data analysis may be more immediately useful.

Projecting a full-screen conventional spreadsheet application, text document, or web browser onto a table is not sufficient to explore how tabletop interaction tech-

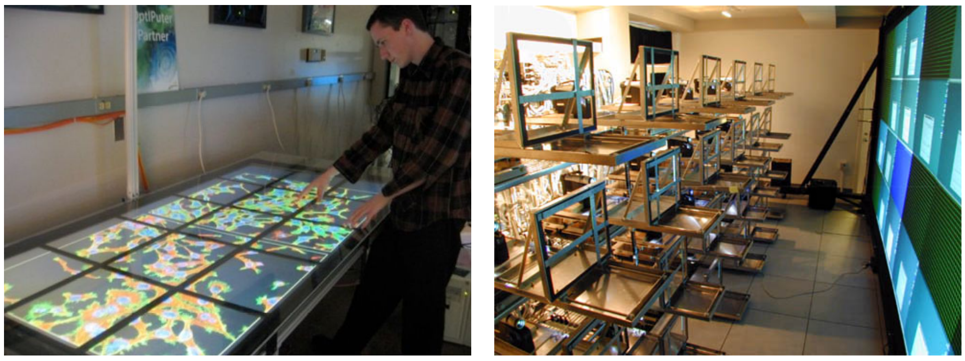


Figure 3.5: Increasing display resolution by tiling: (a) LCD panel displays (from Krumbholz et al., 2005); and (b) projectors (from `http://www.cs.princeton.edu/omnimedia`).

niques can be exploited. Instead, the spreadsheets, documents, or web pages, must be displayed as task artefacts that are small enough to be moved around the table, passed between collaborators, casually grouped into piles and shrunk into special storage task artefacts. This requires the ability to display small text and small user interface components legibly within a large horizontal surface. This in turn necessitates a higher spatial display resolution than many of the tabletop interfaces demonstrated previously.

Consider the resolution required to display legible 12pt text (i.e. the same size as 12pt text from a printer). 1pt is 1/72 inch, so 12pt text is about 1/6inch or 4mm high. A letter “t” at a font size of 12pt in Arial font in fact has a vertical height of around 3mm. (Font size historically corresponds to the height of metal blocks of type and so is larger than the height of the printed characters). To establish a minimum legible font size in pixels, I rendered Arial font into a raster image at different sizes. The minimum legible size is such that a letter t is at least 6px high; parts of the letter merge together at smaller sizes. Accordingly, legible 12pt text requires a minimum resolution of 2px/mm or about 48dpi.

A modest 85cm$\times$85cm table using this resolution requires 2.9Mpx. By contrast, the majority of tabletop interfaces use a single XGA projector with 0.8Mpx or two such projectors with a total resolution of 1.6Mpx. High resolution projectors are expensive and have unsuitable optical properties, and large LCD panel displays cannot be tiled seamlessly because of the bezels (Figure 3.5(a)) (Krumbholz et al., 2005). This is perhaps why such applications have not yet been investigated.

Display resolution can be increased by tiling multiple inexpensive projectors together. Display walls using large numbers of projectors are commonly used for visualisation research (e.g. Li et al., 2000; Guimbretière et al., 2001; Wallace et al., 2005). The Princeton display wall, for instance, uses 24 projectors to create a display of 19Mpx (Figure 3.5(b)) (Wallace et al., 2005). Such displays typically require a cluster of PCs to render the image and a high-bandwidth low-latency network to connect the cluster. Distributed rendering is then performed by specialist software (Cruz-Neira et al., 1992; Humphreys and Hanrahan, 1999; Li et al., 2000; Humphreys et al., 2002; Jeong et al., 2005; Wallace et al., 2005). However, tiling a relatively modest number of projectors can create a display with an appropriate resolution without requiring rendering clusters. The requisite 2.9MPx, for instance, can be achieved using four XGA projectors connected to a single desktop PC using two dual-head graphics adapters.

Tiling multiple projectors in this way is currently both expensive and impractical for large-scale deployment. However, in recent years, the cost of displays has decreased and their resolution increased, and this trend seems likely to continue with the development of new display technologies such as plastic displays. Tiling multiple projectors in the short term therefore enables investigation of interfaces that may become more feasible in the long term.

Tiling multiple projectors introduces two further problems. Firstly, rendering a large number of pixels and task artefacts on multiple graphics outputs reduces performance if approached naïvely. Miede (2006), for instance, showed that an implementation of Interface Currents (Hinrichs et al., 2005) became unresponsive when scaled to a 5Mpx display. Display wall software addresses this problem by exploiting hardware-accelerated rendering on commodity graphics cards to increase performance. The Buffer Framework (Isenberg et al., 2006) additionally uses a novel buffer method to avoid expensive geometric calculations.

Secondly, aligning the projector images is difficult, and so these displays suffer from small overlaps, mismatches and keystoning. Even using purpose-built precision mechanical alignment mountings, aligning four projectors to the required accuracy is time-consuming, requires careful engineering, and often relies on rear projection, which precludes some direct-touch and stylus technologies. Display wall software compensates for projector misalignments by applying small adjustment transformations and blending masks to each frame before it is sent to the projectors (Li et al., 2000). These adjustments are usually calculated using an automatic calibration procedure, avoiding precise mechanical alignment. Ashdown (2004) applied these techniques in a tabletop system to align two projectors to create a 1.6Mpx display.

Most of the established display wall systems reviewed here are designed to use a local cluster of PCs connected by a high-bandwidth low-latency network, rather than to support remote collaboration between sites. Furthermore, none of the systems support tabletop interaction abstractions, and consequently cannot easily be modified to create high-resolution remote tabletop interfaces. Accordingly, although higher resolution tabletop interfaces are both desirable and technically feasible, they cannot easily be created by adapting established systems.

3.3.4 Rapid Prototyping

The widespread use of DiamondSpin (Shen et al., 2004) to create various tabletop applications is partly attributed to its support for rapid prototyping. It enables researchers to reuse existing user interface components such as buttons and file choosers in order to prototype new tabletop applications rapidly. The intention is not to use legacy applications on a large multi-user display, but rather to combine existing components with tabletop interaction techniques. Access to such components avoids the need to reengineer them from scratch for tabletop applications, which reduces the effort required. Complex tabletop applications such as collaborative web-browsing and data analysis become more feasible if existing web-browser or spreadsheet components can be reused. It is not within the scope of most research projects to reengineer such components from scratch for a tabletop environment.

This is especially important when constructing tabletop interfaces that have sufficient spatial display resolution to display such components as task artefacts that may be passed around the table and grouped into containers. Construction of a new tabletop system should accordingly support rapid prototyping of complex tabletop applications through the reuse of existing user interface components.

3.4 Chapter Summary

In spite of a number of research projects, there is little theoretical basis for the design of remote tabletop interfaces, and little discussion of design guidelines for their construction. In this chapter I have considered workspace awareness and tabletop work practices in order to develop design guidelines and questions that will be investigated empirically in Chapter 6. A discussion of technical considerations then established that such a system is feasible but cannot easily be constructed by adapting an existing system. Furthermore, when constructing a new tabletop system, support for existing user interface components and high display resolutions are both desirable and feasible. Lastly, the requirements present three potential performance bottlenecks: limited network bandwidth; rendering many pixels; and rendering on multiple graphics cards. This technical discussion informs the method for constructing high-resolution remote tabletop interfaces, presented in the following chapter.

Chapter 4

Constructing High-Resolution Remote Tabletop Interfaces

This chapter describes a method for the construction of the high-resolution remote tabletop interfaces whose requirements were established in the previous chapter. Ad-hoc technical implementations are often sufficient to support user interface research. In this case, however, a systematic approach is necessitated by the technical challenges described in the previous chapter. Furthermore, in undertaking this work, I aim to enable rapid exploration of this research area, to investigate different interaction techniques and applications. The design of such a system inevitably determines its utility, and this again necessitates such an approach.

Following a brief overview, I describe for each subsystem the possible approaches, the chosen method and implementation issues. The structure of this chapter accordingly considers in turn: the distribution architecture; rendering and display management at the client; user input; arm segmentation; the server and its application programmer interface (API); and finally the reuse of existing user interface components. Each of these methods is then evaluated in the following chapter, along with applications to illustrate the use of the overall system.

4.1 Overview

The system, known as T3, enables high-resolution and remote tabletop applications to be created using a Java API. The application uses the API to create rectangular interactive digital areas known as tiles. The tiles are positioned within a single large coordinate space, and the application can translate, rotate, scale, and change the contents of the tiles in response to user input as desired. In addition to more customised behaviour, tiles can also use existing user interface components in order to enable rapid creation of complex applications. These function as expected without requiring modification, and include standard components such as buttons and text boxes, and also third-party components such as spreadsheets and web-browsers (Figure 4.1(a)).

Each tabletop display is controlled by a local computer running a client program that performs the rendering and receives data from user input devices. Multi-projector displays can be created using multiple multi-head graphics adapters. The client creates the illusion of a single seamless display from loosely-aligned projectors by warping and blending the images sent to each projector (Figure 4.1(b-d)). Multiple clients can connect to the server to create a shared workspace for remote tabletop collaboration, and collaborators' arms can be captured and represented remotely at each site (Figure 4.1(e)).

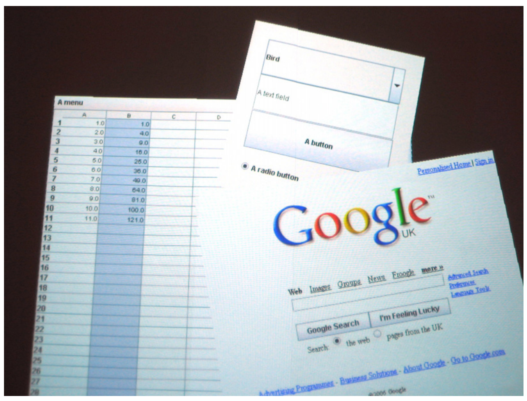
4.2 Distribution Architecture

The ways in which a remote collaboration system is split between different computers greatly influences its characteristics and utility. This section discusses previous approaches and then describes the T3 architecture and implementation issues.

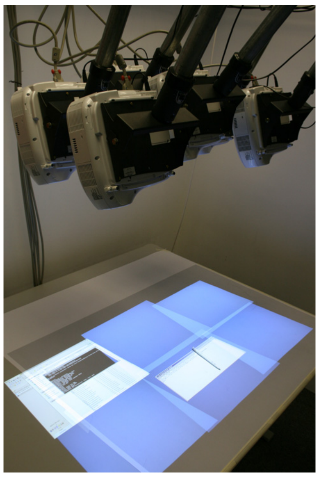
4.2.1 Approaches to Distribution

This section discusses various distribution architectures, which each describe a way of distributing a collaborative application among different computers. Various such architectures were classified and discussed in the early 1990s (e.g. Lauwers et al., 1990; Hill et al., 1994; Patterson, 1995; Phillips, 1999). Much of this work is no longer well known, perhaps because developers have not created the complex distributed systems to which the formalisms are most usefully applied. Nevertheless, the work usefully describes different architectures and their trade-offs.

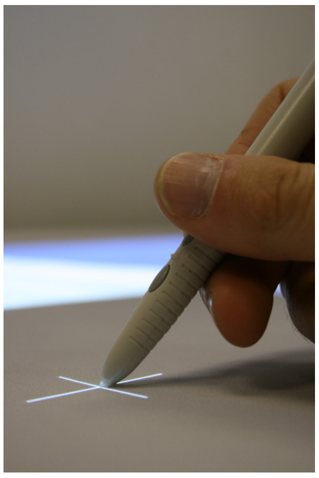
I follow the approach in the literature and consider an application to consist of distinct components. Different components may be situated on different computers, but a component cannot itself be split between computers. The literature describes such components using variants of the Model-View-Controller (MVC) pattern (Gamma et al., 1994): the *model* contains the state; the *controller* manipulates the model based on user input events; and the *view* updates the appearance



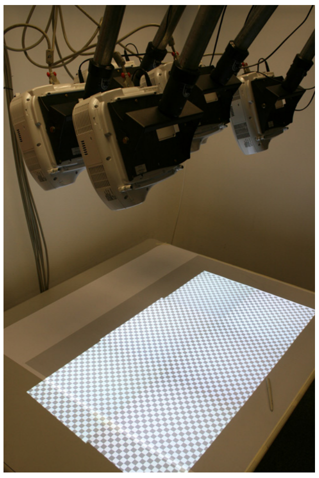
(a)



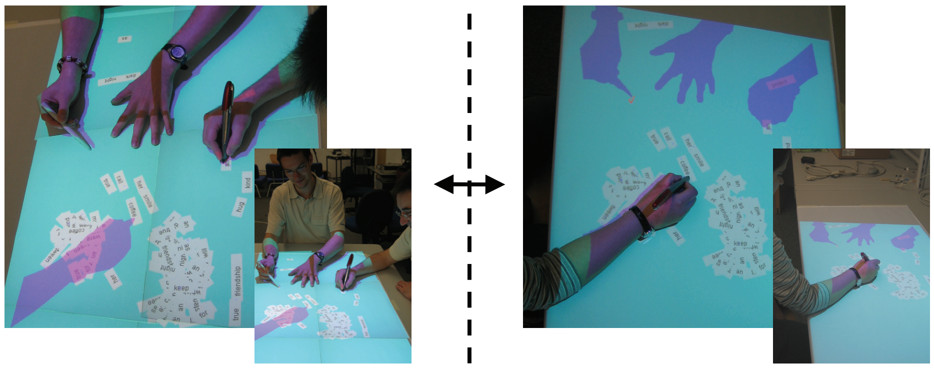
(b)



(c)



(d)



(e)

Figure 4.1: Photos of the system showing: (a) reuse of existing user interface components; (b) the display without correction applied; (c) during registration; and (d) after geometric and photometric correction; and (e) remote collaboration with arm representations.

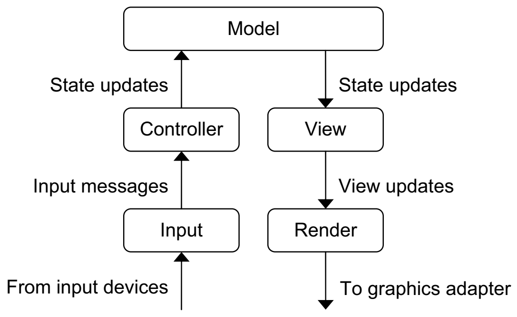


Figure 4.2: An application described using the extended Model-View-Controller components and data flow between.

based on changes in the model. I additionally consider lower-level processing of input events, and rendering (Figure 4.2) (Phillips, 1999).

Two particular constraints apply to the design of the T3 distribution architecture. Firstly, the architecture must exploit abstractions in order to provide a high-resolution shared workspace over the limited bandwidth of a local area network, as described in Section 3.3.

Secondly, support for rapid prototyping through the reuse of existing user interface components prohibits certain architectures. Existing user interface components can be considered as a model, a view and a controller, and are typically accessed using calls to compiled libraries or the windowing system. They cannot easily be split, and so this prohibits architectures that would situate the model, view and controller components on different computers. This leads to three possible architectures.

A *thin-client centralised architecture* system centralises the model, view and controller on a server (Figure 4.3). Each client sends input events to the server, and receives render primitives from the server. Remote display and distributed rendering protocols such as VNC (Richardson et al., 1998), X11 (Scheifler and Gettys, 1996), GLX (Kilgard, 1996), Chromium (Humphreys et al., 2002) and Escritoire (Ashdown, 2004; Ashdown and Robinson, 2005) all use this approach. Thin client systems often exhibit heavy network load because of the potentially large size of the display updates. This can be improved by client caching of state such as the most recent frame (VNC), tile or window images (Escritoire, X11), or texture data (Chromium).

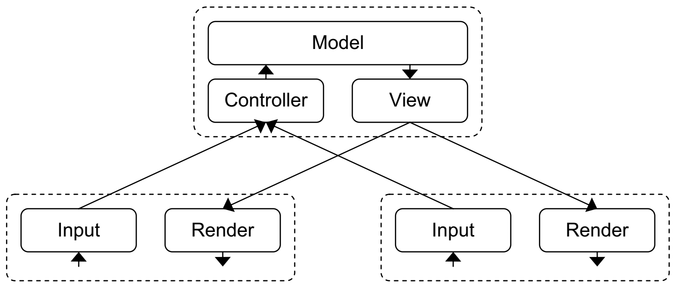


Figure 4.3: Thin-client centralised architecture.

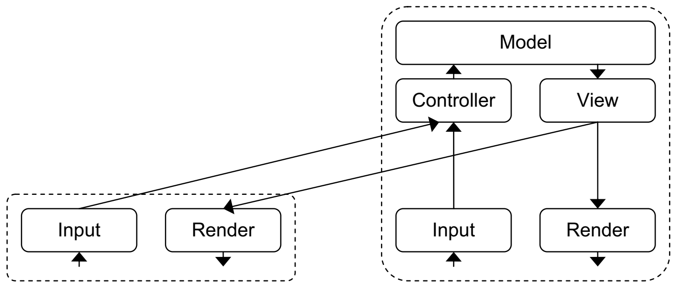


Figure 4.4: Thin-client client-based centralised architecture.

The *thin-client client-based centralised architecture* (Figure 4.4) is a variant in which a single process acts as both client and server, while further clients connect to this process as before. It offers performance gains for one site, but there is less scope for separation of function because the client and server are now integrated. Escritoire, VNC, and X11 demonstrate that conventional centralised architectures can in any case achieve sufficient performance.

The *input-event-synchronizing replicated architecture* replicates all components at each site. Local input events are propagated to every site and injected into the application (Figure 4.5). TIDL (Hutterer et al., 2006) and Digitable (Coldefy and dit Picard, 2007) use input-event-synchronizing replicated architectures. Network load is low because only input events are sent between sites. In spite of its advantages, this approach is difficult to use reliably because it relies on all copies of the application remaining in synchronization while running independently on heterogeneous computers connected by network links with varying delays. If the copies were to lose synchronization then different sites would see different workspaces, and subsequent input would be interpreted differently at the different sites. Lauw-

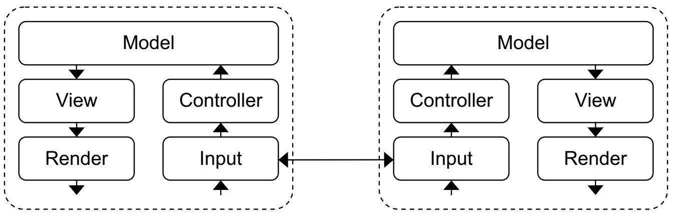


Figure 4.5: Input-event-synchronizing replicated architecture.

ers et al. (1990) discuss the problems of maintaining synchronization. Input events from different sites may be arbitrarily interleaved because of network delays. This can be addressed using a distributed algorithm, or channelling of all events into a single stream through a central server, or a strict floor control procedure. Other problems are less tractable, however. Consider, for instance, opening a new web page in a collaborative web-browsing application. The display state would depend upon the time taken to load and render the page, which will vary between sites. Similar problems arise when task artefacts are animated, such as Interface Currents (Hinrichs et al., 2005), because the precise stage of animation depends on the local CPU load. A slight lag between sites could cause an input event to lie within the task artefact at one site but not at another site. It is not clear how such external non-deterministic elements can be synchronised reliably without imposing severe restrictions on applications. Accordingly, T3 uses a thin-client centralised architecture.

Two further architectures are widely used but inappropriate for existing user interface components because the model, view and controller are split between computers or require modification. The *model-synchronizing replicated architecture* (Figure 4.6(a)) was used in early systems to improve response time despite high network latency by using optimistic algorithms (Lauwers et al., 1990). C-Slate (Izadi et al., 2007) uses this architecture. A *thick-client centralised architecture* (Figure 4.6(b)) uses a centralised model component, which operates as a shared data structure with a conflict resolution mechanism. GroupKit (Roseman and Greenberg, 1996) and VideoArms (Tang, A., et al., 2006a) use this architecture.

4.2.2 Approaches to Thin-Client Centralised Architectures

Within the centralised thin-client architecture, the next choice is the division of rendering functionality between client and server. Table 4.1 shows a variety of

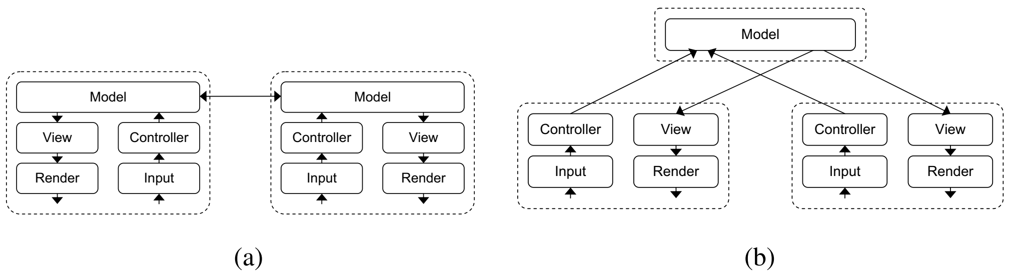


Figure 4.6: Two further architectures: (a) the model-synchronizing replicated architecture; and (b) the thick-client centralised architecture.

	Not Movable-Entity	**Movable-Entity**
Pixel Protocol	VNC	Escritoire, SAGE
Drawing Protocol	Chromium	X11

Table 4.1: Classifications of thin-client centralised architectures.

approaches used by some existing thin-client systems: VNC (Richardson et al., 1998); Escritoire (Ashdown, 2004; Ashdown and Robinson, 2005); SAGE (Jeong et al., 2005); X11 (Scheifler and Gettys, 1996); and Chromium (Humphreys et al., 2002). (X11 uses "server" to designate the process closest to the display, though for consistency I shall refer to this as the X11 client.)

VNC, Escritoire, and SAGE use *pixel protocols* that transmit encoded pixel data to be copied into part of the display, whereas Chromium and X11 use *drawing protocols* that primarily transmit polygons and text (though they also support pixel data). Drawing protocols can exhibit smaller updates and consequently achieve higher frame rates at a given network load, but are more complex. Furthermore, it is difficult to integrate existing user interface components using drawing primitives but, as Section 4.7.3 will illustrate, relatively easy to do so using pixel data.

Escritoire, SAGE, and X11 use *movable-entity protocols* that explicitly describe "windows" or "tiles" that can be moved by transmitting just the new location. VNC and Chromium, by contrast, would require transmission of pixel data or drawing primitives, leading to higher network load and lower frame rates. VNC avoids this problem in practice by supporting the copying of pixel data between specified rectangles in the framebuffer but, as described in Section 3.3.2, this is unlikely to extend to the rotation required by tabletop interfaces. Chromium does not avoid the problem of network load, but assumes that a high-bandwidth low-latency network is available.

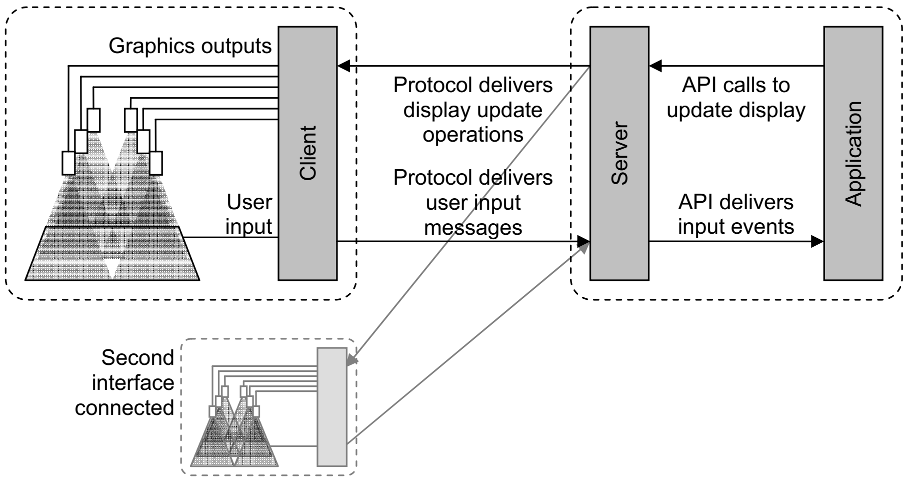

Graphics outputs
User input
Client
Protocol delivers display update operations
Protocol delivers user input messages
Server
API calls to update display
API delivers input events
Application
Second interface connected

Figure 4.7: T3 architecture.

The main factors in choosing an approach are the constraints of limited network bandwidth relative to the large number of pixels, and the reuse of existing user interface components (both identified in Section 3.3). The design guideline of local, direct manipulation (Section 3.1) suggests that many display updates will involve transformation of entities, and so favours a movable-entity protocol. Drawing protocols require less bandwidth than pixel protocols but are more complex to implement from scratch and not easily integrated with existing user interface components. Accordingly, T3 uses a thin-client architecture with a movable-entity pixel protocol, similar to Escritoire.

4.2.3 T3 Architecture

Figure 4.7 illustrates the T3 architecture. The server provides an API used by the applications programmer to create applications. The server translates these API calls into protocol operations that are sent to the clients. Each client performs the rendering for its own multi-projector display. Events generated by input devices at a client are sent back to the server and dispatched to the application through the API.

The protocol from the server to the clients defines movable entities that appear in the workspace. These entities are positioned, independently of the local display configuration, in *surface coordinates* specified in millimetres from the bottom left-hand corner of the display surface. The protocol defines three types of entity:

- A *tile* is a rectangular raster image that appears rotated, scaled and translated onto the display. Tiles are intended to represent the virtual task artefacts typically used in tabletop applications. They can be transformed, repainted, reordered, made visible or invisible, and destroyed.
- A *link* is a line that appears on the display joining two translated and rotated rectangles. Links are intended to represent relationships between task artefacts and parts of task artefacts. The relationship between a hypertext link on a web page and the resulting new web page might, for instance, be illustrated using a link. They can be moved, reordered, recoloured and destroyed.
- A *cursor* is a telepointer trail or arm shadow outline on the display. New points can be added to trails, and the shape of arm shadows can be changed.

The data fields for each entity comprise the current display state. This is stored at each client, for use during rendering, and also at the server, to send to clients joining the session. This state storage is implemented using a *state manager* component. Protocol operations invoked on the state manager alter the stored state and also notify an attached state listener component. The state listener at the server transmits the invoked operations to each client, which then invokes the operations on its own state manager. The state listener at the client tracks the operations invoked from frame to frame in order to determine which projectors need re-rendering.

Figure 4.8 shows this arrangement. API calls from the application are translated by the server into operations invoked on its state manager. The network sender listens and transmits these operations as messages to each client. The network receiver at the client translates these messages back into operations and invokes these on its state manager. The client listens to these operations to determine the display areas to re-render. These areas are then rendered using the data held in the client's state manager.

Input event messages from the clients to the server do not use a state manager component. The T3 protocol supports three types of input device:

- A *keyboard* supports key pressed, released and typed events, which correspond to Java AWT events. A person identifier field allows the server to differentiate multiple keyboards at a single client.

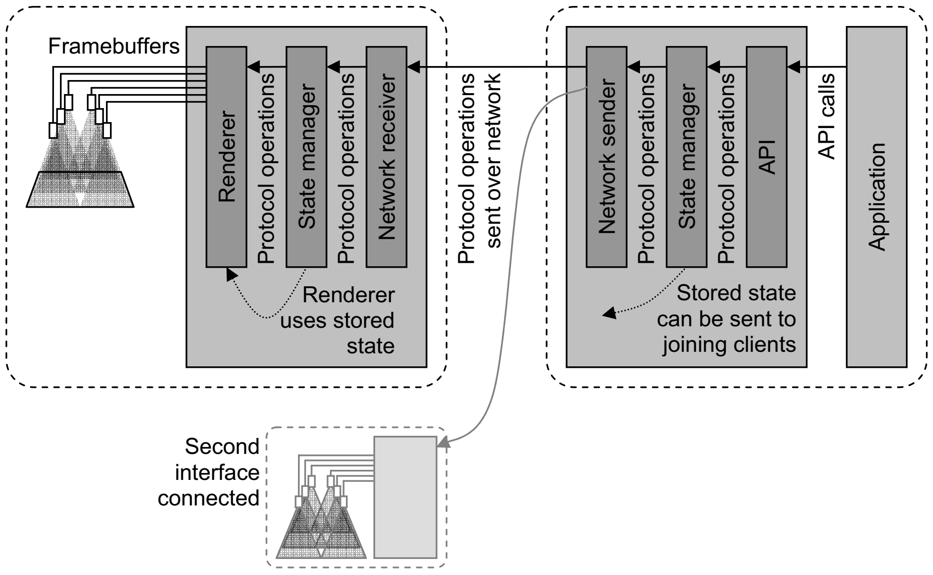


Figure 4.8: T3 implementation using state manager components and state listeners.

- A *point input device* represents its state by a location in surface coordinates and a 32-bit field representing the state of any associated buttons. Additionally, the state can be marked as unknown, or augmented with extra data. Sections 4.4 and 5.7 discuss how this applies to styluses, mice and multi-touch tables.
- An *outline input device*, such as an arm tracking system, generates a set of contours in surface coordinates.

Appendix A describes in more detail the data fields and operations for each type of entity, and the supported messages for each type of input device.

4.2.4 Protocol Implementation

The decoupling of the generation of protocol operations (at the server) from rendering (at the client) presents two problems. Firstly, clients use a standard double-buffering approach to avoid displaying intermediate rendering results, and yet at the server, operations can be generated at any time. Suppose, for instance, that two task artefacts are to appear at the same time as each other. The corresponding two tile visibility messages may be generated back-to-back at the server but processed between different frames at the client, with the result that one task artefact appears before the other. The server addresses this problem by explicitly grouping messages into *framebuffer updates*, such that the client applies all messages within a single framebuffer update while rendering the same frame. The division between framebuffer updates in the message stream is marked with a framebuffer update message.

Secondly, operations at the server can be invoked at any time with relative ease, but are processed at the client at a rate depending on its rendering performance and network bandwidth. This rate also varies between clients. A series of rapid updates may therefore overwhelm a low-performing client or network link if the server transmits all operations as they are invoked. VNC (Richardson et al., 1998) addresses this using a client-pull scheme, in which the server stores messages from successive framebuffer updates in a client-pull buffer. Instead of sending to clients immediately, it awaits an explicit request from the client and then sends all messages as a single framebuffer update. Messages in the client-pull buffer often supersede each other and are coalesced to reduce the total size and the processing time required at the client. The protocol therefore adapts so that clients with low-performing hardware or network links will receive updates of a similar size but less frequently. One client-pull buffer must be maintained at the server for each client.

When using a client-pull scheme in a remote collaboration system, however, it is necessary to repeatedly poll the server in order to receive new messages soon after they are generated. T3 uses the adapted client-pull scheme developed by Ashdown (2004) to addresses this problem. If a client requests messages and none are buffered, then the system switches to server-push mode, in which all messages in the next framebuffer update are transmitted to the client as soon as they are generated. The system then switches back to client-pull mode, and the client must explicitly request further updates.

Like VNC, T3 coalesces messages in the client-pull buffer:

- When a destroy message is added to the buffer, any messages in the buffer relating to that entity are removed.
- When tile visibility, tile transformation, link transformation, cursor trail point or cursor outline messages are added to the buffer, any previous such messages relating to the same entity are removed.
- At most one tile and link order message is ever sent in a burst.

- Tile repaint messages are removed from the buffer if subsumed by a later repaint message.

T3 uses a TCP connection between the server and each client, and messages are represented as serialized Java objects. Large tile image data may be compressed using run-length encoding.

4.3 Display Management

Having established the client-server protocol, I now describe how the client renders the image based on the data held in its state manager. This process addresses the problem of producing an image that appears seamless and correct in spite of loosely-positioned projectors. This is considered as two sub-problems:

- *Geometric errors* arise because of keystoning, misalignments and overlaps. The windows in Figure 4.1(b) (page 81), for instance, do not appear rectangular because of the keystoning.
- *Photometric errors* arise in areas where projectors overlap, and because of colour variation between and within projectors. Overlapping areas in Figure 4.1(b) (page 81), for instance, appear much brighter than other areas.

Each of these two problems is addressed by estimating the distortion using a registration procedure, and then correcting by manipulating the framebuffer image for each projector in such a way as to compensate for this modelled distortion. Figure 4.1(c and d) (page 81) show the registration process and the corrected display. The rendering process must also address the problem of rendering a large number of pixels at a reasonable frame rate on multiple graphics outputs. This section considers in turn geometric correction, geometric registration, photometric correction, and various aspects of the implementation.

4.3.1 Geometric Correction

I begin by defining more formally the aims and process of geometric correction, using the coordinate space approach of Ashdown (2004). Consider the three coordinate spaces shown in Figure 4.9:

- Image data for each tile is described in its tile (T) coordinate system, defined in pixels.

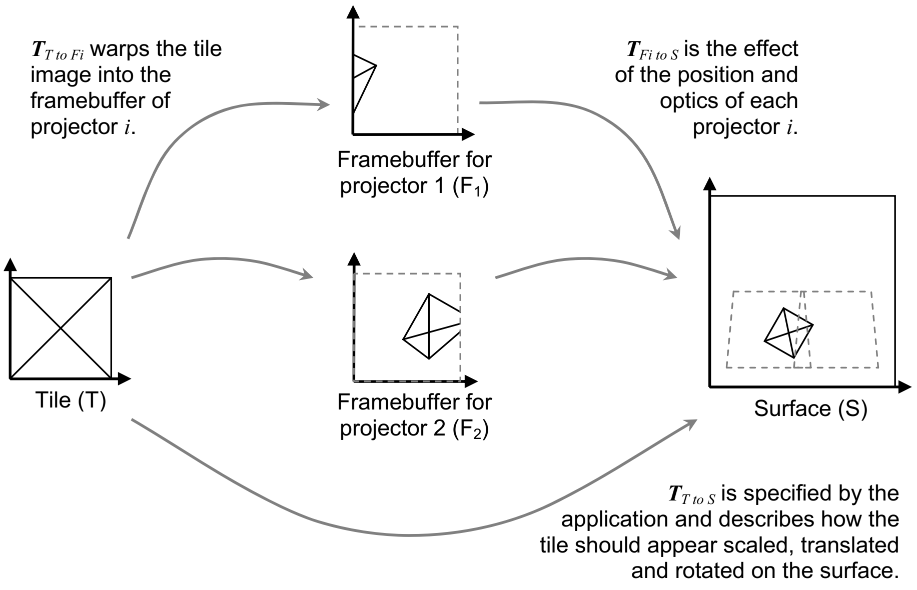

$T_{T\ to\ Fi}$ warps the tile image into the framebuffer of projector i.
Framebuffer for projector 1 (F$_1$)
$T_{Fi\ to\ S}$ is the effect of the position and optics of each projector i.
Tile (T)
Framebuffer for projector 2 (F$_2$)
Surface (S)
$T_{T\ to\ S}$ is specified by the application and describes how the tile should appear scaled, translated and rotated on the surface.

Figure 4.9: Coordinate spaces and transformations for geometric correction.

- The surface (S) coordinate system is expressed in millimetres from the bottom-left corner of the display surface.
- The framebuffer (F$_i$) coordinate system corresponds to the framebuffer pixels for the graphics output for projector i.

The arrows in Figure 4.9 illustrate the transformations between these coordinate spaces. $\boldsymbol{T}_{T\ to\ Fi}$ describes how the system renders the tile into the framebuffer of projector i, such that the pixel at coordinates $\boldsymbol{x}$ in the tile appears at coordinates $\boldsymbol{T}_{T\ to\ Fi}.\boldsymbol{x}$ in the framebuffer. $\boldsymbol{T}_{Fi\ to\ S}$ describes how locations in the framebuffer of each projector i map to locations on the display surface. This is governed by the mechanical positioning and optics of the projector. Consequently, location $\boldsymbol{x}$ in the tile appears on the surface at coordinates $\boldsymbol{T}_{Fi\ to\ S}.\boldsymbol{T}_{T\ to\ Fi}.\boldsymbol{x}$ when rendered by projector i.

The process of geometric correction computes $\boldsymbol{T}_{T\ to\ Fi}$, for each projector i, to satisfy two constraints. Firstly, if the displayed image is to appear correct, then a given location $\boldsymbol{x}$ in a tile image must appear at the same location on the surface regardless of which projector displays it. Accordingly, each $\boldsymbol{T}_{T\ to\ Fi}$ must be set so that the system renders the tile into each framebuffer to satisfy, for every i and j, $\boldsymbol{T}_{Fi\ to\ S}.\boldsymbol{T}_{T\ to\ Fi}.\boldsymbol{x} = \boldsymbol{T}_{Fj\ to\ S}.\boldsymbol{T}_{T\ to\ Fj}.\boldsymbol{x}$. Therefore, for all i, $\boldsymbol{T}_{Fi\ to\ S}.\boldsymbol{T}_{T\ to\ Fi} = \boldsymbol{C}$, for some transformation $\boldsymbol{C}$.

Secondly, the application specifies how the tile should appear rotated, translated and scaled on the display surface. It does this by specifying $\boldsymbol{T}_{T\ to\ S}$, which describes how a given location $\boldsymbol{x}$ in the tile should appear on the surface at location $\boldsymbol{T}_{T\ to\ S}.\boldsymbol{x}$. This gives rise to the second constraint: for all i, $\boldsymbol{T}_{Fi\ to\ S}.\boldsymbol{T}_{T\ to\ Fi} = \boldsymbol{T}_{T\ to\ S}$.

Geometric correction typically proceeds, for each projector i, by first estimating $\boldsymbol{T}_{S\ to\ Fi}$ using a geometric registration procedure, described in more detail later in this section. ($\boldsymbol{T}_{S\ to\ Fi}$ is defined as the inverse of $\boldsymbol{T}_{Fi\ to\ S}$.) The two above constraints are then satisfied by setting $\boldsymbol{T}_{T\ to\ Fi} = \boldsymbol{T}_{S\ to\ Fi}.\boldsymbol{T}_{T\ to\ S}$. The type of mathematical transformation used to model $\boldsymbol{T}_{S\ to\ Fi}$ depends on the particular geometric correction method being used. Majumder and Brown (2007) review various methods for planar display surfaces.

The simplest method assumes that $\boldsymbol{T}_{S\ to\ Fi}$ is a linear parametric transformation between homogeneous coordinates or, equivalently, that the projector is the inverse of a pin-hole camera. In this case, $\boldsymbol{T}_{S\ to\ Fi}$ is a 3×3 matrix with eight degrees of freedom, known as a planar homography, which maps homogeneous coordinates in the surface coordinate space to homogeneous coordinates in the framebuffer. It is computed from point correspondences between the framebuffer and the surface using a procedure described later in this section. These correspondences are determined using a geometric registration technique described in the following section. This linear parametric approach has been widely investigated and implemented in multi-projector toolkit systems (e.g. Yang et al., 2001; Humphreys et al., 2002; Wallace et al., 2005). It can be implemented efficiently by carrying out the resulting $\boldsymbol{T}_{T\ to\ Fi}$ using the texture warping functions found in commodity graphics adapters. Relatively cheap projectors and short-throw projectors can, however, demonstrate significant non-linearities due to lens distortion, leading to errors at the projector boundaries when using the linear parametric approach. (Camera lenses, sometimes used during calibration, can also exhibit non-linearities, though this can be addressed using camera-calibration implementations such as the Camera Calibration Toolbox for Matlab).

Majumder and Brown (2007) review two other geometric correction methods for planar surfaces, which model $\boldsymbol{T}_{S\ to\ Fi}$ either as a two-dimensional cubic polynomial transformation (Hereld et al., 2002), or a piecewise linear transformation (Brown and Seales, 2002). Both approaches have efficient implementations using texture mapping hardware, by warping textures onto cubic surfaces (cubic), and by splitting into a triangular mesh and warping each triangle using an affine transformation (piecewise linear). These implementations are more complex and less efficient than the linear parametric approach. Both mechanisms achieve sub-pixel accuracy in the presence of limited lens distortion.

T3 adopts a pragmatic approach, and aims not to support further research into multi-projector correction techniques but rather to allow HCI researchers to rapidly create prototype high-resolution tabletop interfaces for investigation. Accordingly, T3 implements the relatively simple linear parametric correction. In spite of its limitations, this nevertheless supports investigation of interaction techniques and collaboration in a range of high-resolution applications (Section 5.6). If the technique is found in practice to be too limiting then the more advanced techniques can be accommodated without making significant changes to the rendering architecture.

4.3.2 Geometric Registration

Geometric registration is the process of determining, for each projector i, the transformation $\boldsymbol{T}_{S\ to\ Fi}$ between locations on the surface and locations in the framebuffer. For the chosen linear parametric correction method, this is a 3×3 matrix mapping between homogenous coordinates. The transformation is computed using the following method described by Hartley and Zisserman (2000, page 91). The method requires a set of locations (X_j, Y_j) in framebuffer coordinates (F$_i$) and the corresponding locations (x_j, y_j) in surface coordinates (S). It requires a minimum of four of these initial correspondences per projector, and uses a least-squares method to calculate more accurate results given a larger number of correspondences.

Prior work has used various techniques to capture these initial correspondences: automatically using a camera (Raskar et al., 1999); or using multiple cameras (Chen, H., et al., 2002); or semi-automatically if the display surface is an accurate high-resolution input device such as a digitising tablet (Ashdown, 2004). The tablet-based approach requires work by the user, who must touch the stylus on a series of projected points, and may require subsequent intervention to correct small errors. Nevertheless, registration is expected to take place infrequently, and the tablet-based approach avoids the need to mount a tree of cameras. All the installations in this dissertation used a high-resolution stylus input, and so accordingly T3 uses a tablet-based approach to obtain correspondences.

The transformation is then computed as follows:

Let:

$$\boldsymbol{T}_{S\ to\ Fi} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix}$$

and so:

$$\begin{pmatrix} X'_j \\ Y'_j \\ W'_j \end{pmatrix} = \boldsymbol{T}_{S\ to\ Fi} . \begin{pmatrix} x_j \\ y_j \\ 1 \end{pmatrix}$$

where:

$$(X_j, Y_j) = (\frac{X'_j}{W'_j}, \frac{Y'_j}{W'_j})$$

This can be written:

$$X_j = \frac{h_{11}x_j + h_{12}y_j + h_{13}}{h_{31}x_j + h_{32}y_j + h_{33}}$$

$$Y_j = \frac{h_{21}x_j + h_{22}y_j + h_{23}}{h_{31}x_j + h_{32}y_j + h_{33}}$$

or equivalently:

$$\begin{pmatrix} x_0 & y_0 & 1 & 0 & 0 & 0 & -x_0X_0 & -y_0X_0 & -X_0 \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1X_1 & -y_1X_1 & -X_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_nX_n & -y_0X_n & -X_n \\ 0 & 0 & 0 & x_0 & y_0 & 1 & -x_0Y_0 & -y_0Y_0 & -Y_0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1Y_1 & -y_1Y_1 & -Y_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & x_n & y_n & 1 & -x_nY_n & -y_nY_n & -Y_n \end{pmatrix} . \begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{pmatrix} = \boldsymbol{0}$$

or equivalently, letting $\boldsymbol{A}$ and $\boldsymbol{h}$ be the left-hand matrix and right-hand vector respectively:

$$\boldsymbol{A}.\boldsymbol{h} = \boldsymbol{0}$$

This method seeks a non-zero solution for $\boldsymbol{h}$. Scaling equally all the elements of vector $\boldsymbol{h}$ has no effect on the transformation itself, because the resulting X', Y' and W' coordinates will be equally scaled, and thus the resulting coordinates $(X'/W', Y'/W')$ remain unchanged. The magnitude of vector $\boldsymbol{h}$ can therefore be constrained to be 1.

$\boldsymbol{h}$ has an exact solution if $n = 4$, and a least squares solution when $n > 4$. The solution is calculated by computing the single value decomposition of $\boldsymbol{A}$, $\boldsymbol{A} = \boldsymbol{U}.\boldsymbol{D}.\boldsymbol{V}^T$, and taking $\boldsymbol{h}$ to be the last column of $\boldsymbol{V}$. $\boldsymbol{T}_{S\ to\ Fi}$ is then constructed from the elements of $\boldsymbol{h}$.

4.3.3 Photometric Correction and Registration

Photometric correction models the colour distortion within- and between- projectors, and in the overlap regions, and then manipulates the framebuffer image to produce a seamless overall image. Majumder and Brown (2007) review various techniques.

Blending techniques (e.g. Raskar et al., 1999) use geometric registration to identify areas where projectors overlap, and then apply alpha masks to the framebuffers to ensure a smooth transition between projectors in these areas. They have a simple implementation but do not correct for either form of colour variation, though they help to make the effect less perceptible. Gamut-matching techniques (e.g. Stone, 2001) use a precise light measuring device to measure and correct for colour variation between projectors, but do not alone address the problem of overlap. Majumder and Stevens (2004) measure colour variation within and between projectors and in overlap areas using a camera, and then correct the framebuffer images in a way that can be implemented efficiently using pixel shaders.

T3 again adopts a pragmatic approach, implementing the blending technique. This does not address the problems of colour variation, but the evaluation in the following chapter will show that, though perceptible, colour variation is sufficiently small that it is not distracting in practice.

Some notation must be introduced before the alpha mask generation procedure is described. Consider a location in surface coordinates, $\boldsymbol{x}$, at which two projectors overlap. Suppose $\alpha(i, \boldsymbol{x})$ is the alpha mask value for projector i at this location, where $0 \leq \alpha(i, \boldsymbol{x}) \leq 1$. Assuming perfect geometric correction and without any blending, then in any given colour channel, both projectors would project the same intensity $p(\boldsymbol{x})$ at this point. This is weighted by their individual alpha mask values, so that the total intensity appearing on the surface at $\boldsymbol{x}$ is therefore:

$$t(\boldsymbol{x}) = p(\boldsymbol{x}).\alpha(1, \boldsymbol{x}) + p(\boldsymbol{x}).\alpha(2, \boldsymbol{x})$$

The goal of blending is to set the alpha mask $\alpha(i, \boldsymbol{x})$ for each projector i to satisfy two constraints. Firstly, it is desirable that $t(\boldsymbol{x}) = p(\boldsymbol{x})$ for all points $\boldsymbol{x}$, so that at all points, the total displayed intensity is the same as that intended by the application. This requires $\alpha(1, \boldsymbol{x}) + \alpha(2, \boldsymbol{x}) = 1$ for all points $\boldsymbol{x}$. This is somewhat intuitive and extends to an arbitrary number of projectors: $\sum_i \alpha(i, \boldsymbol{x}) = 1$ for all points $\boldsymbol{x}$. Secondly, in order to avoid visible seams, it is desirable to vary the alpha masks smoothly in the overlap regions in order to blend smoothly between projectors.

I use the blending procedure described by Raskar et al. (1999) and reviewed by Majumder and Brown (2007). Let $dist(i, \boldsymbol{x})$ be the Euclidean distance in the

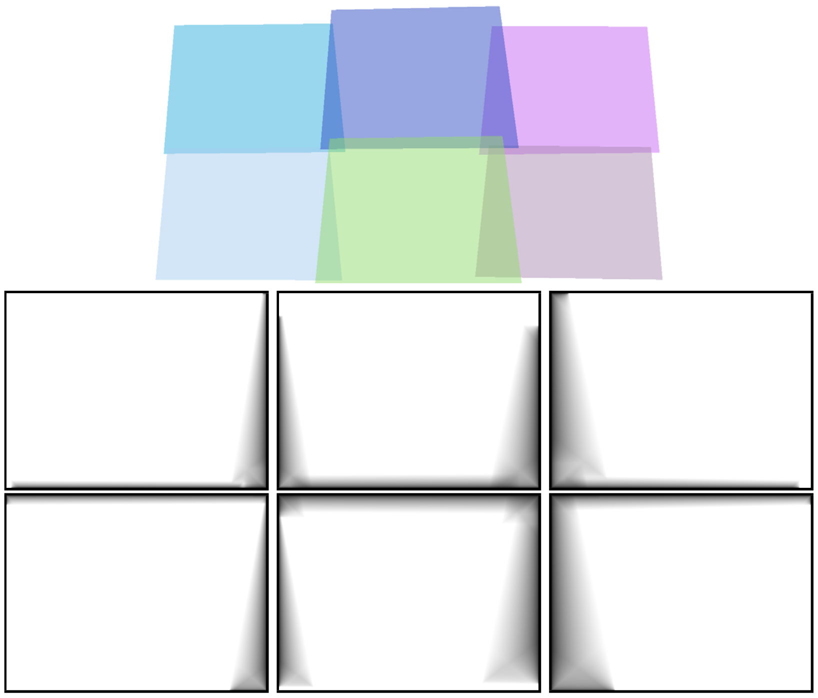


Figure 4.10: Projector arrangement and resulting alpha masks.

surface coordinate space of point $\boldsymbol{x}$ from nearest edge of the extent of projector i. The two projector case can achieve smooth blending between the two images by defining $\alpha(i, \boldsymbol{x})$ in the overlap region as:

$$\alpha(1, \boldsymbol{x}) = \frac{dist(1, \boldsymbol{x})}{dist(1, \boldsymbol{x}) + dist(2, \boldsymbol{x})}$$

$$\alpha(2, \boldsymbol{x}) = \frac{dist(2, \boldsymbol{x})}{dist(1, x) + dist(2, \boldsymbol{x})}$$

This extends to points outside the overlap region by defining $dist(i, \boldsymbol{x})$ to be zero when location $\boldsymbol{x}$ lies outside projector i. It also extends to arbitrary numbers of projectors:

$$\alpha(i, \boldsymbol{x}) = \frac{dist(i, \boldsymbol{x})}{\sum_m dist(m, \boldsymbol{x})}$$

Having defined $\alpha(i, \boldsymbol{x})$, the value of the alpha mask for projector i at a given pixel is given by $\alpha(i, \boldsymbol{T}_{Fi\ to\ S}(\boldsymbol{y}))$, where $\boldsymbol{y}$ is the framebuffer coordinates of the centre of the pixel. Figure 4.10 shows a projector layout and the resulting alpha masks.

4.3.4 Implementation and Performance

In addition to geometric and photometric correction, the client must also render a large number of pixels onto multiple graphics outputs. This is a potential performance bottleneck, as identified in Section 3.3. On a multi-projector display with no parallelism, the time taken to render the display is equal to the product of the number of projectors to be rendered and the time taken to render each projector frame. This section describes the rendering implementation and the measures taken to achieve performance. This is then evaluated in the following chapter.

Procedure for Rendering a Frame

Rendering a frame for a particular projector, i, involves rendering the tile images, links and cursors that appear on the projector in order into the framebuffer. Unlike some multi-projector display systems (e.g. Ashdown, 2004; Majumder and Brown, 2007), geometric correction is applied as the entities are rendered, avoiding the overhead of applying geometric correction retrospectively.

Links, trail cursors and outline cursors are rendered for projector i by applying $\boldsymbol{T}_{S\ to\ Fi}$ to the surface coordinates of their component lines and polygons specified by the server. (Outline cursors are not necessarily convex, and must be converted to a triangle mesh prior to rendering, using a tessellator).

The tile-rendering process exploits the texture-rendering hardware found on commodity graphics adapters. The tile image is maintained in texture memory on the graphics adapter to prevent the graphics bus from becoming a bottleneck. A tile is rendered for projector i as follows. $\boldsymbol{T}_{T\ to\ Fi}$ is first calculated by multiplying the matrices $\boldsymbol{T}_{S\ to\ Fi}$ (computed during the geometric registration procedure) and $\boldsymbol{T}_{T\ to\ S}$ (specified by the server and held in the state manager). This transformation is applied to the tile's corner points in tile space to determine their coordinates in the framebuffer. The graphics hardware then renders the tile from texture memory into this quadrilateral in the framebuffer.

Blending is performed once the entities have been rendered. The alpha mask for each projector is stored as an image in texture memory and applied pixel for pixel to the framebuffer.

```
Loop forever:

    Check input devices for new events and send to server

    If any data is available from the network:

        While more data is available without blocking,
        and a framebuffer update has not yet been received:

            Read another message from the network.

        If a new framebuffer update message has been received:

            Send a ready message to server.

            For each buffered message:
                Translate the message into a protocol operation.
                Invoke the operation on the local state manager.

            For each projector whose contents will have changed:
                Cause its opengl window to be rerendered
                synchronously (*).
                If sufficient time has elapsed since
                input devices were last checked:
                    Check input devices for new events and
                    send to server.

            For each projector whose contents has changed:
                Swap the buffers of its OpenGL window.
```

Figure 4.11: Pseudocode showing the control loop of the client renderer thread.

This process is implemented using OpenGL and Java. The most frequent matrix calculations and geometric intersection operations are performed on demand, and the results cached until invalidated. Quadrilateral intersections are approximated using bounding boxes where appropriate.

Procedure for Rendering on Multiple Projectors

The process begins by creating a full-screen OpenGL window in the appropriate graphics output for each projector. A single control loop renders successive frames following the procedure in Figure 4.11.

In order to improve performance, the system selectively re-renders only those projectors whose contents will have changed. This is implemented by listening to the operations invoked on the client's state manager and comparing each to the area covered by each projector.

Double-buffering prevents intermediate rendering results from being displayed. The buffer-swaps for all projectors are synchronized as far as possible so that the projectors appear consistent with each other.

I also implemented a multi-threaded approach with the aim of improving performance by using parallel rendering. Each projector had a thread which would perform the rendering step (marked (*) in Figure 4.11) in parallel and then synchronize with the other threads using a thread barrier. Initial tests indicated no performance increase. This suggests that the underlying device drivers and hardware performs rendering either sequentially or in parallel, regardless of how the calls are invoked by the program.

Tile images in texture memory must in general be stored and updated separately on different graphics outputs. Storing every tile image in texture memory on every graphics output does not require manipulation of texture memory when a tile is translated from one projector to another. Alternatively, the tile image can be stored only on the graphics outputs on which the tile is visible. This second scheme minimises the use of texture memory and the number of data transfers required to repaint a tile, at the cost of moving the texture data between different graphics outputs when the tile moves around the display. T3 supports both methods. Large high-resolution tiles that span several projectors will not perform well under either method and are transparently split by the client into smaller tiles.

4.4 Input Devices

This section briefly describes the supported input devices. The point input device, keyboard device and associated state, were introduced in Section 4.2.3.

T3 supports large graphics tablets using the standard Wintab software interface. Device state is marked as unknown when the stylus is lifted, and each stylus button corresponds to a single bit of the button field. Large commercially-available graphics tablets, however, support only a single stylus on each tablet. Styluses using Anoto technology overcome this limitation (as described in Section 2.4). T3 supports the Maxell DP 201 Anoto stylus. The stylus streams coordinates to the client PC over Bluetooth using a serial byte-stream protocol, which is then interpreted by the T3 client. The device state is again marked as unknown when the stylus is lifted. The Anoto styluses have no buttons, and so the buttons field is always zero. Bluetooth in Windows XP supports only a single serial connection at any time (even using multiple adapters). In order to support several styluses on the same surface (perhaps for co-located collaborators or bimanual techniques), T3 requires additional computers in the vicinity to receive the additional Bluetooth signals and send the corresponding updates to the server. This work-around

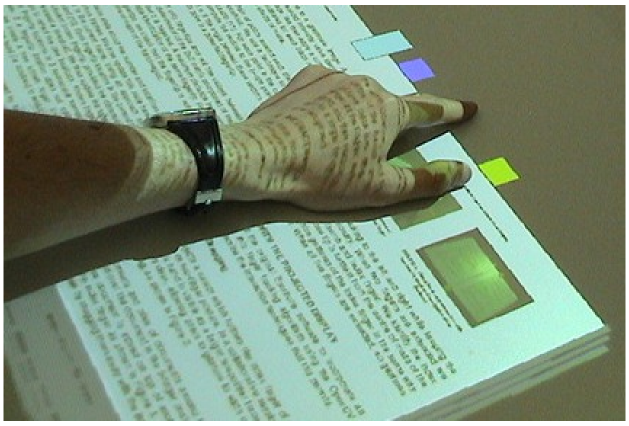


Figure 4.12: Discolouration of the arm using a front-projected display.

is appropriate for a laboratory prototype system. Finally, mouse support allows development and debugging on systems without direct input devices.

Each T3 client supports only a single keyboard, because the lower-level device interfaces to support multiple keyboards in Windows XP seem incompatible with full-screen Java OpenGL windows. Multiple keyboards are supported using a similar workaround to the Anoto styluses, and so collaborators sitting around the surface are each able to enter text without having to share a single keyboard.

4.5 Arm Segmentation

This section describes how T3 captures the outline of collaborators' arms to support remote arm representations.

4.5.1 Approaches

Large displays limit the scope to apply many of the object tracking and arm segmentation methods that researchers have presented in the interaction technology and computer vision literature. The changing display varies the background from which the arm is to be segmented. Front projection introduces further problems, because the projected light discolours the hand, limiting the scope to apply colour-based segmentation or background modelling (Figure 4.12). Rear projection is prohibited by many direct input devices commonly used in tabletop research, such as capacitive sensing surfaces (e.g. Dietz and Leigh, 2001) and, in particular, the stylus systems available during the course of this research.

Koike et al. (2001) overcome these problems of segmenting arms in front-projected systems by using a sensitive passive infra-red camera, without an infra-red light source. This enables reliable segmentation because the arm emits more infra-red light than the display surface. Such passive infra-red cameras are expensive (£4k at the price of writing) compared to the projectors. A similar approach using a front-mounted infra-red light source and a cheaper front-mounted camera with an infra-red-only filter is unlikely to enable arm segmentation: Wilson (2005) used a similar hardware configuration to infer points of contact, and his figures suggest that the infra-red light intensities from the arms and display surfaces are too similar for reliable segmentation.

Several methods instead use commodity front-mounted cameras to segment arms in front-projected systems. Methods using background subtraction based on luminance (von Hardenberg and Bérard, 2001) or colour (Letessier and Bérard, 2004) maintain a model of the background image against which each video frame is compared pixel by pixel to identify changes, which are assumed to be caused by the presence of the arm in that location. Tang, A., et al. (2006a) demonstrated a colour-based segmentation method that identifies areas of skin-colour without a background model. Neither background subtraction methods nor colour-based segmentation methods model the display contents, and so both can be impaired by hand discoloration. They have been demonstrated in practice by constraining the display to mitigate this problem: discolouration can be reduced by limiting the display intensity and constraining the display contents to be mainly black; and constraining the remaining areas to be mainly white ensures that discolouration affects luminance but not hue.

A more recent background-subtraction method continuously updates the background model using the current display contents itself (Coldefy and dit Picard, 2007). The background model is a computer-generated approximation to the camera image with no arms present, and is created by applying geometric and photometric transformations to the framebuffer. Transformation parameters are estimated using a calibration process. This method is in principle robust to changing display contents and, with a more accurate background model, may be less affected by hand discolouration. However, empirical results show that the segmentation is nevertheless not yet entirely reliable. Furthermore, background image generation is computationally intensive and runs on a high-end consumer computer at 17fps at a resolution of 160px $\times$ 128px. It is not clear how this would scale to a multi-projector system with multiple framebuffers.

4.5.2 Arm segmentation in T3

Conventional background-subtraction methods using front-mounted commodity cameras remain most feasible. However, the constraints on display content necessary to avoid hand discolouration and changing background are not always appropriate. The prohibition of large areas of colour, for instance, limits the scope to investigate large movable container task artefacts.

I have developed a variant of the background-subtraction approach, that constrains the display in ways that may be more appropriate for remote tabletop interfaces. Although the red and green display components may vary, the blue component is constrained to be uniform across the entire display, except for very small areas such as black text. Arms are then segmented from the background using background subtraction in the blue colour plane (Figure 4.13).

This method assumes that arms reflect less blue light than the display surface. This is true if the skin colour is much darker than the display surface. Additionally, for white skin, which is slightly pink and therefore a relatively poor reflector of blue light, the assumption holds if the display surface is at least as light as the skin colour and also a shade of grey. The method further assumes that the camera can discern the difference, and so cameras with small dynamic range may require reduced display intensity. Dark shadows cast by the arm are segmented together with the arm, though in practice the projectors are positioned so that the shadows fall close to the arm.

The background model (Figure 4.13(c)) corresponds to the blue component of the first frame captured, and is periodically updated using a weighted average to account for temporal lighting variations. Each frame is processed by subtracting from its blue component the background model, pixel by pixel. A large difference corresponds to a strong likelihood that the pixel in question is part of an arm. The difference is thresholded using a configurable preset (Figure 4.13(d)). Camera pixels lying outside the display surface are ignored. The algorithm computes the set of contours from the thresholded image. Small contours are filtered out to remove noise, and the remaining contours form the segmentation result. Contour points are transformed from the camera coordinate frame into surface coordinates using a linear transformation computed during a semi-automatic calibration procedure similar to that described in Section 4.3.2.

4.5.3 Implementation

The implementation uses a number of optimisations. The algorithm is implemented in C using the OpenCV library, rather than slower Java. It runs as a sepa-

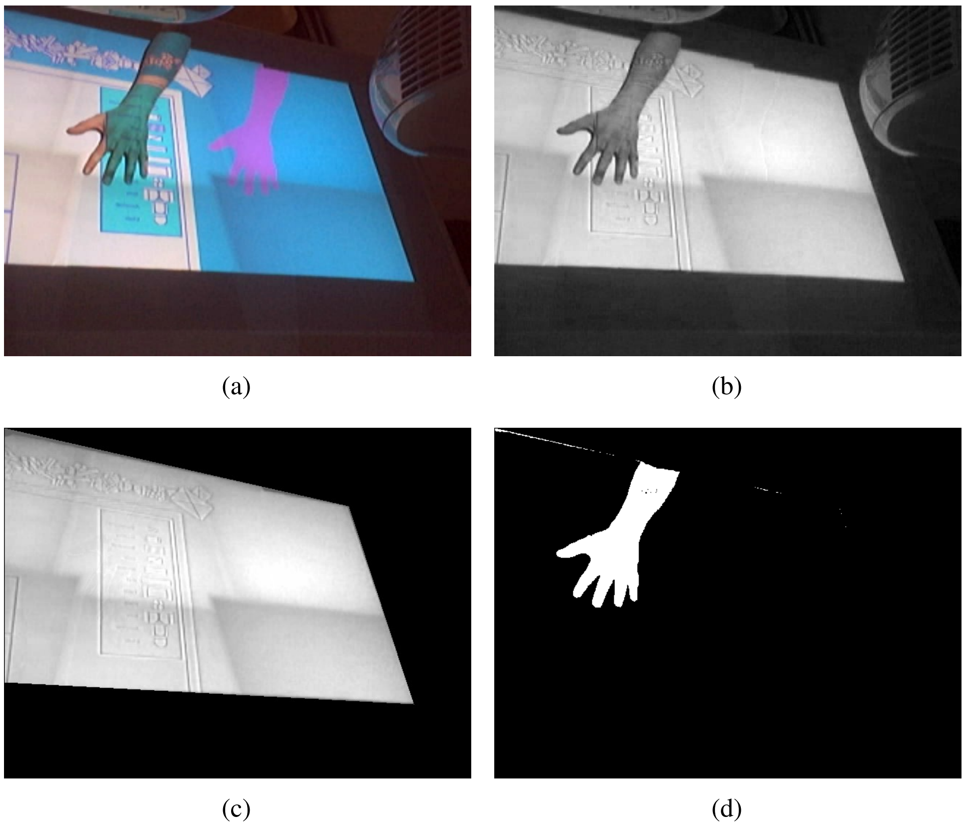


Figure 4.13: Stages in the arm segmentation algorithm: (a) the camera view; (b) the same view showing only the blue component; (c) the background model; and (d) the thresholded image.

rate process and outputs the contours to its standard output stream, which is read regularly by the T3 client. The stream protocol uses a length field for each frame so that the client may quickly skip over any accumulated data from old frames. The T3 client reads the stream into a large circular buffer in chunks to minimise the number of operating system calls. Contour data in protocol messages is stored in the byte-stream format outputted by the image processing application, in order to avoid data marshalling. The volume of data is reduced in the byte-stream and in protocol messages by approximating contour data to a polygon using the Douglas-Peucker algorithm. Points are expressed in 4-byte image coordinates instead of 16-byte surface coordinates.

4.6 Server and API

Having described the rendering, input and arm segmentation methods at the client, I now describe the server. As shown in Figures 4.7 (page 86) and 4.8 (page 88), the server fulfils two roles.

Firstly, the server augments the underlying protocol abstractions, such as tiles, to provide a more useful programming interface for the application programmer. The T3 protocol entities, such as tiles, present a well-defined set of operations, but were motivated primarily by performance and architectural considerations and serve primarily as a thin-client protocol. The API, on the other hand, aims to provide abstractions that reduce the complexity required to create applications. It sits between the application and the state manager at the server, so that API calls invoked by the application are translated into protocol operations and invoked on the state manager. Input messages received from clients are translated into API events and passed to the application.

Secondly, the server implements a concurrency architecture that enables it to respond to messages from multiple connected clients, to listen for new clients, and to process asynchronous operations from the application such as animations.

This section describes firstly the design of the API, and secondly the implementation of the concurrency architecture.

4.6.1 API Design

The API provides several features intended to reduce the complexity required to create applications. These are based on the API features provided by existing tabletop systems such as DiamondSpin (Shen et al., 2004), remote tabletop systems such as Escritoire (Ashdown, 2004), and conventional windowing systems.

Object-Oriented Programming

The server is implemented in Java. Tiles are represented in the server API by instances of the Portfolio class, which augments their functionality in various ways, described throughout this section. The Portfolio class, for instance, has several abstract methods to handle input events and to repaint the tile, which must be overridden by the application programmer in a subclass. Abstracting to a class enables application programmers to encapsulate functions and state, and to create reusable templates for visual elements that can be extended and instantiated multiple times. The naming distinction between tiles and portfolios serves to highlight the difference between API and protocol, and follows the convention of Ashdown (2004).

Input Delivery

Protocol messages from clients indicate input device state such as surface coordinates and buttons, but applications respond to *changes* in such state. The course of action then often depends on the visual element on which the event occurs, and on the coordinates of the event in the frame of reference of this element.

The T3 server accordingly translates input messages from clients into input state change events, which are delivered to the event-handling methods of the appropriate portfolios. By default, this corresponds to the front-most tile on the surface at the event coordinates. The API presents event coordinates in the frame of reference of the surface or of a specified tile. Portfolios can also register to receive all input events, either temporarily or indefinitely. This enables dragging techniques in which the stylus may move temporarily outside the moving tile. Portfolios are also notified when an input device leaves or enters a tile.

Regrouping of Visual Elements

Many tabletop applications allow task artefacts to be dragged into a container task artefact and subsequently moved as a group, as discussed in Section 2.2.3. Explicit group abstractions avoid the need for the application programmer to manipulate each contained task artefact individually, and a group coordinate space enables relative positioning of the contained task artefacts.

The T3 API provides grouping and hierarchy using the Portfolio class. Each portfolio has a parent portfolio and may have many child portfolios. A portfolio may have a tile or may be a grouping element with no tile. Each portfolio has a coordinate system, defined relative to that of its parent using a scale, rotation and translation, which determines the location of its tile. Consequently, when a portfolio is moved, its own tile and the tiles of all of its children move with it. Visibility also applies to groups so that a tile is visible if, and only if, its portfolio is visible and each of its ancestor portfolios is also visible. The ordering from back to front of the tiles on the surface depends on the grouping of portfolios and the ordering of each portfolio's children, and is determined by walking the tree of portfolios. Input events may be passed from child to parent portfolio, enabling, for example, a group to be dragged by dragging one of its component tiles.

Multi-Threading and Locking

The server internally uses a multi-threaded architecture. The API provides two mechanisms for the application programmer to perform operations without having to explicitly manage threading, locking, or marking of framebuffer updates:

- *Synchronous operations* are invoked within an event-handling routine written by the application programmer that, in turn, has been called by T3 in response to an input event. Before calling the routine, T3 ensures that the calling thread has the necessary locks on shared data structures. T3 marks the framebuffer update as complete once the input event has been fully processed.

- *Asynchronous operations* may be initiated by the application programmer at any time from any thread. They might occur, for example, to create portfolios when an application starts, or to animate the movement of portfolios. The API provides a procedure for carrying out such operations. The executable sequence of operations to be carried out asynchronously is passed to this procedure as a Java Runnable object. This procedure then ensures that the sequence is executed with the correct locks and that a framebuffer update is marked as complete once the sequence has finished. An animator class enables repeated asynchronous execution at a particular frame rate.

Rendering

The API provides a procedure to aid repainting of a specified part of a portfolio's tile. It creates an off-screen raster image, and then calls the portfolio's overridden repaint method to paint into that image. The method then invokes a tile repaint protocol operation to send the resulting image to the connected clients.

4.6.2 Concurrency Architecture

Approaches

Besides presenting the API, the server must also respond to messages from connected clients, listen for new clients, and process asynchronous operations such as animations. There are three possible widely-used concurrency architectures when implementing such a system:

- *One thread per client and one listener thread.* The thread for each client repeatedly blocks awaiting the next protocol message from that client, and then processes that message and any operations invoked in response to it. A further thread listens for new clients. Asynchronous operations can be carried out by additional threads running at lower priority levels. A locking mechanism is required to protect shared data structures.

- *One thread per client, one executor thread and one listener thread.* The thread for each client repeatedly reads protocol messages from that client and adds them to a central event queue. Messages on this queue are then processed by a single executor thread. Asynchronous operations are added to the queue as Runnable objects. No locking is needed beyond the event queue structure.

- *One thread for all clients, and one listener thread.* A single thread processes buffered events for each client in turn in a round-robin fashion. Asynchronous operations are added as Runnable objects to a special buffer. No locking is needed.

The latter methods are more difficult to integrate with existing windowing systems, which typically have their own thread constraints. Java Swing operations such as repainting, for instance, must run in a particular thread. This is easily accommodated in the first method, whereas the other methods require such operations to be processed first by the windowing system thread and then by the executor thread, with a potential degradation in responsiveness. The first method also allows additional threads such as animators to be scheduled at a low priority to ensure they do not compromise responsiveness.

Implementation

The T3 server accordingly uses the first method, and each client thread proceeds as follows. When a protocol message is received from a client, the corresponding thread at the server firstly identifies it as arising from either a point input device, a keyboard, or an outline input device (described in Section 4.2.3). For point and outline input devices, which have corresponding cursors, the thread invokes an operation on the state manager to update the location of the corresponding cursor at the clients. For keyboard and point input devices, the message is then compared to the most recent message from that input device to determine input state change events, such as whether buttons have been pressed. The same thread processes each of these events by identifying the appropriate portfolio and calling its event-handling method. Once all events have been processed, the thread marks the framebuffer update as complete, causing any generated protocol messages to be sent through to the clients, or buffered in a client-pull buffer. It then blocks awaiting the next protocol message.

T3 uses a simple locking scheme in which a single lock protects all shared data structures, such as the state manager and the portfolio objects. Some windowing systems use more fine-grained schemes, such as locking per window (e.g. Nelson, 1991). In T3, however, the dependencies between portfolios and their ancestors limit the scope for concurrency. Furthermore, the processing of protocol messages generally involves calculations and copying between buffers rather than activities, such as reading files, that might cause a thread to block. Concurrency therefore offers limited scope to increase performance. This simple locking scheme avoids overheads of fine-grained locking when there is no contention and, in the following chapter, will be shown to perform adequately in the presence of contention from animations.

The emphasis on direct manipulation, localised to the vicinity of the hand, suggests that much server activity will involve moving portfolios and links, and these operations are optimised accordingly. Geometric calculations are performed on demand and cached until invalidated. Tile repaint operations are optimised to avoid copying of large image buffers. Image buffers are also reused where possible to avoid potentially time-consuming allocation and deallocation.

4.7 Reuse of Existing User Interface Components

In this section I discuss how to address the problems of supporting the reuse of existing conventional user interface components, such as buttons and web-browser components. This presents two technical challenges: addressing the input and output differences between conventional user interface components and tabletop interaction; and distributing input events and changes in appearance between these components and the sites for remote collaboration. This section considers prior approaches to each problem and describes the T3 implementation.

4.7.1 Differences in Input and Output Models

Tabletop interfaces differ from the input and output models used by conventional user interface components:

- *Multi-user input.* Tabletop interfaces support multiple users interacting concurrently, whereas conventional components assume a single user.

- *Bimanual and gestural input* devices used by tabletop systems do not map well to the mouse-based input model used by conventional components, which represent input state by a single coordinate pair and a field indicating the button state.

- *Input lifting.* Direct input devices used in tabletop systems typically cease to stream coordinates once a hand or stylus is lifted from the surface. Unlike the mouse, there is not a continuous stream of coordinates as the focus of interaction changes from one location to another.

- *Rotation and scaling* of task artefacts in tabletop applications requires rendering and event processing mechanisms that are not widely present in conventional components.

Various systems address these problems. Multi-Pointer X (Hutterer and Thomas, 2007) and TIDL (Hutterer et al., 2006) enable concurrent interaction by multiple users in unmodified X Windows applications and Java Swing components respectively. DiamondSpin (Shen et al., 2004) and Digitable (Coldefy and dit Picard, 2007) additionally provide rotation and scaling of Java Swing components. Cruiser (Apted et al., 2005) and Escritoire (Ashdown, 2004; Ashdown and Robinson, 2005) use the VNC remote display protocol (Richardson et al., 1998) to provide a multi-user tabletop interface to legacy applications. All these systems use similar approaches to address or avoid the identified problems:

- *Multi-user input* is partially addressed by aggregating all users' event streams. This single stream is then injected into the user interface components. This method enables multiple users to interact provided they use social protocol to take turns. If, however, they interact simultaneously then the component sees a single cursor that jumps between disparate states, which can cause problems with dragging, double clicking, and entering and exiting components. This problem cannot be further resolved without modification of the components.

- *Bimanual and gestural input* is generally not supported. Users are constrained to single-handed input using single-fingered pointing actions, to provide a single coordinate pair suitable for conventional user interface components.

- *Input lifting* is ignored, so again the component sees a mouse cursor jump between disparate states when a pen or finger is lifted and then placed back on the surface.

- *Rotation and scaling* is either avoided or is implemented by capturing the raster output of component rendering. This is then transformed as it is rendered into the framebuffer. Inverse transforms are applied to the coordinates of input events before they are injected into the components.

Existing systems use one of two mechanisms to capture output and transform input. Firstly, Cruiser and Escritoire use the VNC remote display protocol (Richardson et al., 1998) to capture the raster output of rendering the component. These systems then render this image rotated, scaled and translated into the framebuffer as desired. Input events in this framebuffer region are inverse-transformed before being injected into the VNC input stream. DiamondSpin (Shen et al., 2004) and Digitable (Coldefy and dit Picard, 2007) use a second mechanism, which exploits the Java Swing windowing system. When a Swing component is redrawn, these systems paint the component into an off-screen image buffer, which is then rendered rotated, scaled and translated into the framebuffer. The Swing input event system is altered to apply inverse transforms to input events.

The first mechanism is less appropriate when using existing user interface components as building blocks for larger tabletop applications, because the components are isolated from other visual elements on the surface. They cannot, for instance, be manipulated in response to users' interactions with other visual elements on the surface, and vice versa.

4.7.2 Distributing Events and Updates

The mechanism by which changes in appearance and input events are distributed between the components and the sites depends on the distribution architecture. Input-synchronizing replicated systems such as TIDL (Hutterer et al., 2006) capture input device updates at a low level and distribute them to all sites, where they are injected into the event stream for the components. Changes in appearance then occur at each site and need not be distributed. Thin-client centralised architectures must capture the components' changes in appearance, but need not distribute input events. Capturing render output for distribution can be accomplished using the same methods as for rotating and scaling: the VNC mechanism used by Escritoire (Ashdown, 2004; Ashdown and Robinson, 2005) and Cruiser (Apted et al., 2005); and the Java Swing mechanism used by DiamondSpin (Shen et al., 2004) and Digitable (Coldefy and dit Picard, 2007).

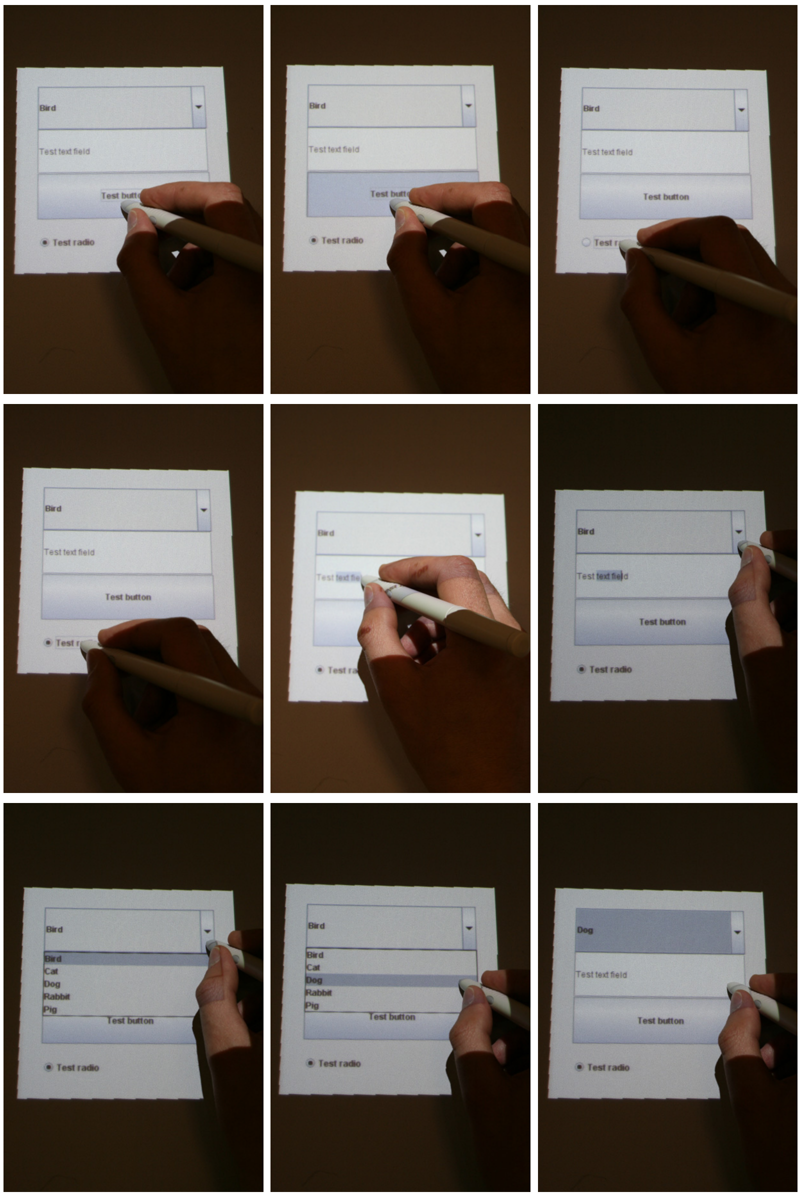


Figure 4.14: Demonstration of reuse of basic user interface components in T3.

4.7.3 Reuse of Existing User Interface Components in T3

T3 uses the Java Swing approach for injecting input events and capturing changes in appearance. Application programmers can create portfolios whose tiles correspond to Swing windows. Unmodified Swing components, such as buttons, web-browsers, and spreadsheets can be placed into these windows. The appearance of the tiles reflects that of the windows, and tiles can be rotated, scaled, translated and made visible and invisible as before. Figures 4.1(a) (page 81) and 4.14 show examples.

Events received by such a portfolio are translated into Swing events and injected into its window at the appropriate location. T3 handles differences between input models in the same way as other systems, and with the same limitations: event locations are represented by single coordinate pairs; events from multiple users and devices are aggregated into a single stream; and no events are injected when the stylus or hand is lifted from the surface. Other portfolios can be manipulated from the event handling routines of the Swing components, and vice versa, and so Swing components can be used as building blocks to create larger tabletop applications.

4.7.4 Implementation

The Swing windows are created at the server as internal windows within a large "multiple-document" parent window using the Swing JDesktopPane class. T3 renders these windows to off-screen image buffers, rather than to the screen, and so the number and size of the windows is not limited by the available screen space on the server computer.

The rendering mechanism uses two stages: firstly, capturing the coordinates of the rectangular regions of windows that must be repainted; and secondly, rendering these regions into off-screen image buffers to be sent to clients. Capturing of coordinates of regions to be repainted, known as *dirty regions*, is achieved by modifying the Swing repaint manager system. This is performed at run-time by sub-classing the RepaintManager class. Whenever a Swing component requests to render a region of itself, the modified repaint manager stores the coordinates of the region.

Later, when Swing initiates repainting, the modified repaint manager triggers the second part of the mechanism. T3 uses the stored component region coordinates to infer the portfolio regions to be rerendered. It creates appropriate off-screen raster images and calls the Swing repaint methods of the corresponding windows, passing as arguments the rendering viewports into these raster images. The regions of the windows are rendered accordingly into the raster images, which are then sent to the clients as described previously. Once rendering is complete, the server clears the list of stored regions.

T3 input events in Swing portfolios are translated into Java AWT event objects. Coordinates are translated into the frame of reference of the appropriate window, and the event is injected into the Swing component deepest in the component layout tree at that point. Keyboard events are dispatched to the Swing component with which the person interacted most recently. In practice this enables multiple collaborators, each with their own keyboard, to type simultaneously into different Swing portfolios.

Care is required to avoid deadlock in the implementation of the modified Repaint-Manager, and the integration of the T3 and Swing synchronisation mechanisms. Appendix B describes the problems and measures taken to address them.

4.8 Chapter Summary

In this chapter I have presented a method for the construction of high-resolution remote tabletop interfaces. I have discussed the prior work and implementation issues for each subsystem: the distribution architecture; display management; input devices; arm segmentation; the server and API; and the reuse of existing desktop user interface components. The following chapter will evaluate the performance and utility of the resulting system.

Chapter 5

Technical Evaluation

In this chapter I present an evaluation of the technical work presented in the previous chapter, in order to establish the limitations and utility of the method. I begin by describing the apparatus used to create the interactive surfaces, and then evaluate each of the sub-methods, considering in turn the geometric and photometric correction techniques; the performance of the server, client, and network protocol; arm segmentation and representation; and the reuse of existing user interface components. I then describe several research projects undertaken by myself, and by others, using the T3 implementation, and conclude with a discussion of the method as a whole.

5.1 Apparatus

I created three tabletop displays on which to test the method:

Display A. Six XGA projectors (1024px×768px) were connected to a single PC via three dual-head graphics adapters to create a display of 4.7Mpx (Figure 5.1). The display area was adjustable between 1100mm×600mm and 1250mm×700mm, using the optical zoom of the projectors. The smallest area provided an average spatial resolution of 2.56px/mm vertically and 2.80px/mm horizontally, assuming that the overlaps between projector images were small. This exceeds the 2.0px/mm minimum required for 12pt text established in Section 3.3.3. The projectors were loosely-aligned and front-mounted using scaffolding and conventional mounting plates. The central projectors were slightly recessed to avoid the air exhaust of one projector feeding the air intake of the next. A conventional web-cam (Logitech QuickCam Pro 4000) was mounted using an Arri

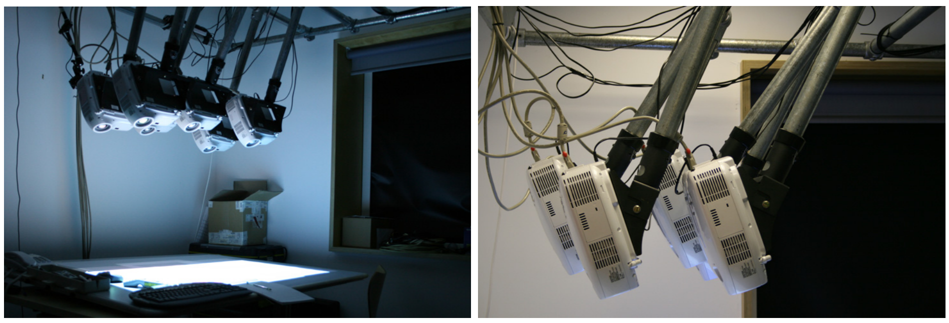


Figure 5.1: Constructed display A.

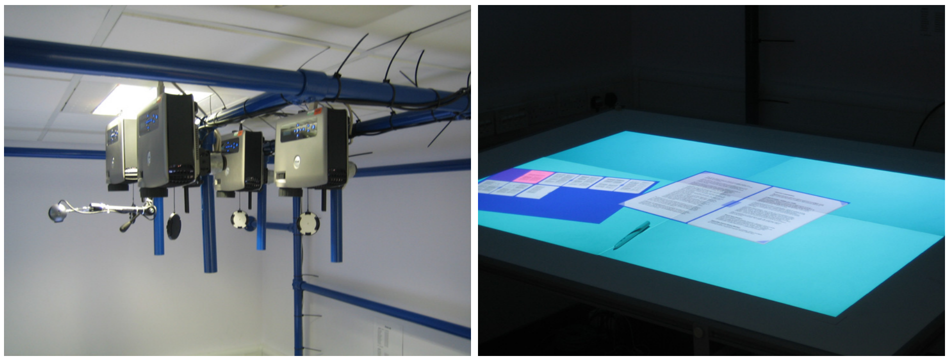


Figure 5.2: Constructed display B.

Magic Arm. It used two stylus input mechanisms: a GTCO CalComp SummaGrid V digitizing tablet with an active area of 1189mm×841mm (A0 size) and a single stylus with two barrel buttons; and also two Hitachi Maxell DP-201 Anoto styluses, with a similar active area. The PC used an Intel Core 2 6600 processor running at 2.4GHz with 1GB RAM. The display was driven by three NVIDIA GeForce 7600GT adapters connected to the Intel D975XBX2 motherboard using PCI-Express slots running at 16x, 8x and 4x respectively.

Display B. Four SXGA+ projectors (1400px×1050px) were connected to a single PC via two dual-head graphics adapters to create a display of 5.7Mpx (Figure 5.2). The display area was approximately 1189mm×841mm (A0 size), providing a spatial resolution of 2.35px/mm horizontally and 2.49px/mm vertically. These were again both higher than the 2px/mm minimum. Stylus input again was provided using a GTCO CalComp SummaGrid V digitizing tablet. The projector mounting was similar to display A, and the PC hardware was the same as that of display A. A conventional web-cam (Logitech QuickCam Pro 4000) was mounted using an Arri Magic Arm.

Display C. A single XGA projector was connected to a single PC to create a display of 0.8Mpx. The display area was approximately 1000mm$\times$1000mm, providing a spatial resolution of approximately 0.9px/mm. Stylus input was provided using the same Anoto styluses as display A. The projector was front-mounted using scaffolding and conventional mounting plates, and a conventional web-cam (Logitech QuickCam Pro 4000) was mounted using an Arri Magic Arm. The PC used an Intel Pentium 4 3.2GHz processor with 1GB RAM and an NVIDIA PCI GeForce FX 5200 graphics adapter.

The software ran successfully on each display. The performance tests described in this chapter were carried out on display A. The T3 server ran on a second PC using an Intel Pentium 4 3.2GHz processor with 1GB RAM and connected using 100Mb/s Ethernet.

5.2 Geometric and Photometric Correction

The implementation uses a semi-automatic geometric calibration procedure, described in Section 4.3.2, in which the user presses the stylus onto projected points in turn.

In practice, 9 points per projector yielded adequate results while still taking under 5 minutes to press on the points for 6 projectors. Small errors sometimes arose from human error when positioning the stylus and from non-linearities in the lenses, and sometimes required subsequent manual correction by editing a configuration file. Alternative techniques may be necessary on larger displays, or in the presence of greater lens distortion, or in the absence of a high-resolution stylus system. In these cases it may be necessary to use registration techniques such as multiple cameras and structured light (Chen, H., et al., 2002), and correction techniques such as cubic polynomial or piecewise linear transformations (reviewed in Sections 4.3.1 and 4.3.3).

Figure 5.3 shows the effects of alpha blending. Unlike the more advanced camera-based techniques described in Section 4.3.3, this procedure does not compensate for colour variation within or between projectors. Nevertheless, in practice these effects were not found to be distracting.

The geometric and photometric correction procedures both affect the legibility of projected text. Firstly, geometric correction warps rectangular tiles into quadrilaterals in the framebuffer and causes blurring of high-contrast features such as

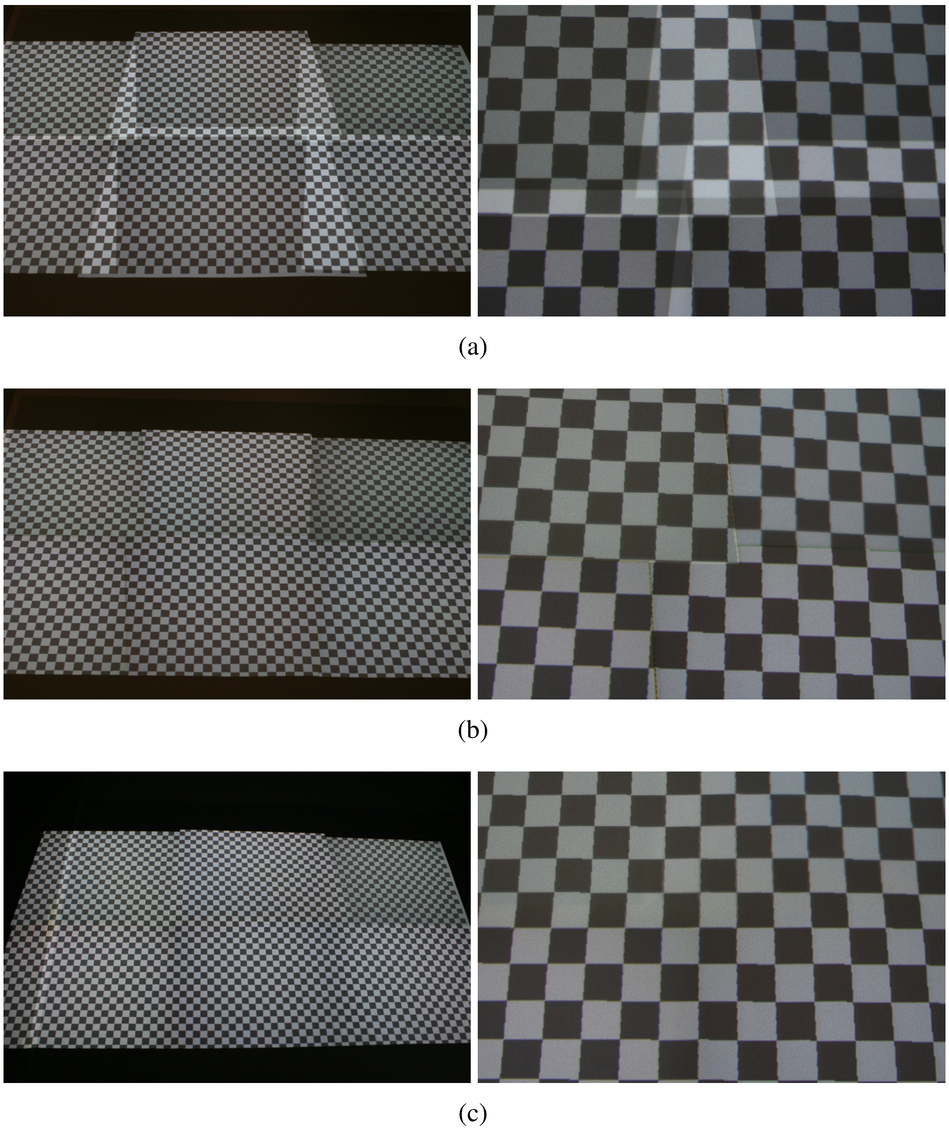
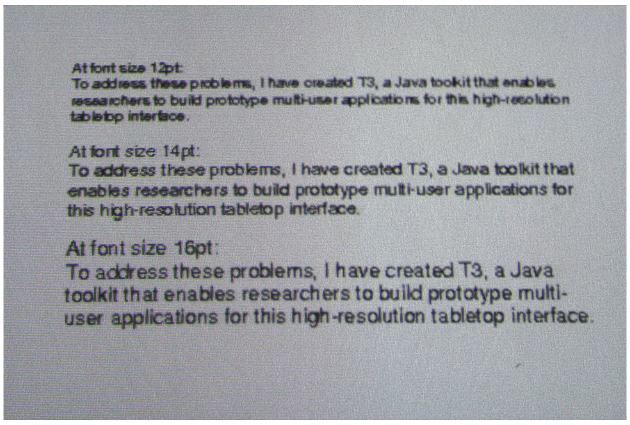


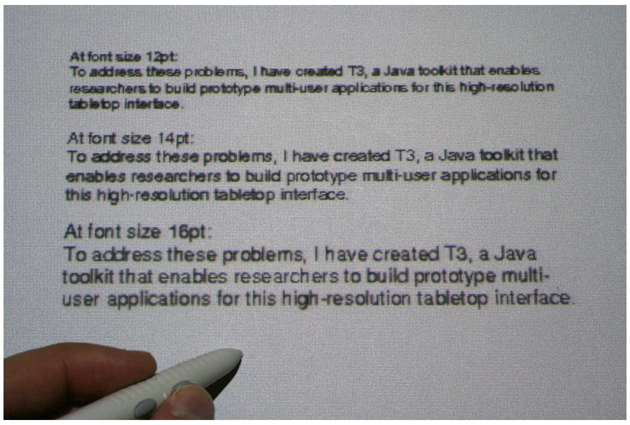
Figure 5.3: Geometric and photometric correction on display A: (a) without photometric correction, overlaps are visible; (b) blacking-out of overlap regions leads to visible seams; (c) using alpha blending, seams are no longer visible. (The white stripe on the left-hand side of the display marks the edge of the Anoto paper.)


At font size 12pt:
To address these problems, I have created T3, a Java toolkit that enables researchers to build prototype multi-user applications for this high-resolution tabletop interface.

At font size 14pt:
To address these problems, I have created T3, a Java toolkit that enables researchers to build prototype multi-user applications for this high-resolution tabletop interface.

At font size 16pt:
To address these problems, I have created T3, a Java toolkit that enables researchers to build prototype multi-user applications for this high-resolution tabletop interface.

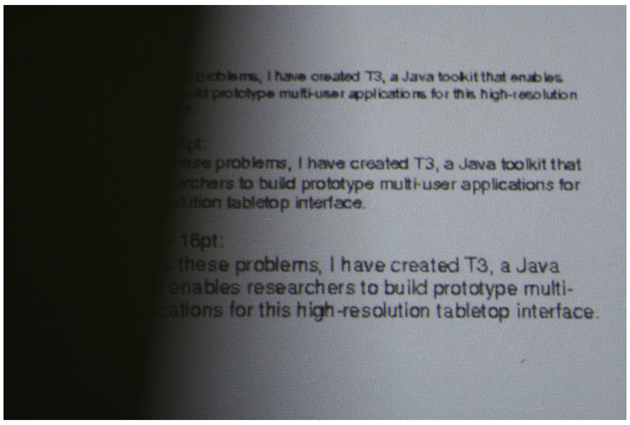
(a)


At font size 12pt:
To address these problems, I have created T3, a Java toolkit that enables researchers to build prototype multi-user applications for this high-resolution tabletop interface.

At font size 14pt:
To address these problems, I have created T3, a Java toolkit that enables researchers to build prototype multi-user applications for this high-resolution tabletop interface.

At font size 16pt:
To address these problems, I have created T3, a Java toolkit that enables researchers to build prototype multi-user applications for this high-resolution tabletop interface.

(b)


problems, I have created T3, a Java toolkit that enables
prototype multi-user applications for this high-resolution

pt:
se problems, I have created T3, a Java toolkit that
rchers to build prototype multi-user applications for
ution tabletop interface.

16pt:
these problems, I have created T3, a Java
enables researchers to build prototype multi-
ations for this high-resolution tabletop interface.

(c)

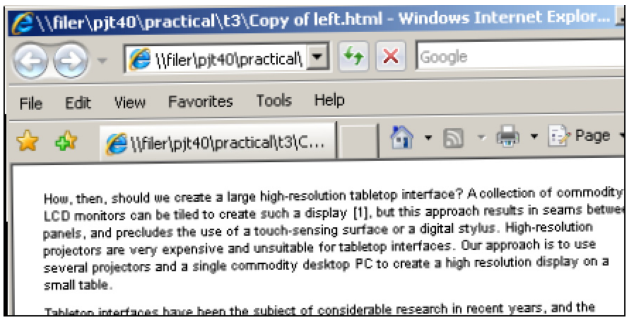
Figure 5.4: Text on display A: (a) within a projector; (b) spanning two projectors; and (c) spanning two projectors with one projector switched off.

text. T3, like most conventional multi-projector display systems, uses bilinear interpolation to perform this warping. The colour of each framebuffer pixel is determined by transforming its centre point into tile space and then calculating a weighted average of the colours of the four tile pixels surrounding this point. This averaging blurs the high-contrast black-white boundary features of on-screen text, decreasing legibility. In practice, 12pt text remains barely legible (Figure 5.4(a)).

Secondly, alpha blending assumes that overlapping pixels align exactly in size, location and orientation. In practice, this is not the case because of keystoning, non-linearities, and small orientation misalignments. This potentially causes further blurring in overlap regions, though in practice this is much less perceptible than the blurring effects of bilinear interpolation (Figures 5.4(b) and 5.4(c)) .

Hereld and Stevens (2005) proposed a novel image-warping technique to address the blurring problem of bilinear interpolation. They argued that bilinear interpolation compromises text crispness for the sake of positioning to the accuracy of fractional pixels. They proposed an alternative technique in which the image is considered an “ocean” of warpable content in which there are “islands” of high-contrast unwarpable content. Their technique identifies high-contrast elements in the source image, such as words or lines of text. These are then copied without interpolation, using a “cookie-cutter” operation, into the interpolated image. This results in a warped image in which photographs and other low-contrast features are warped using bilinear interpolation, while text and other high-contrast features appear crisp.

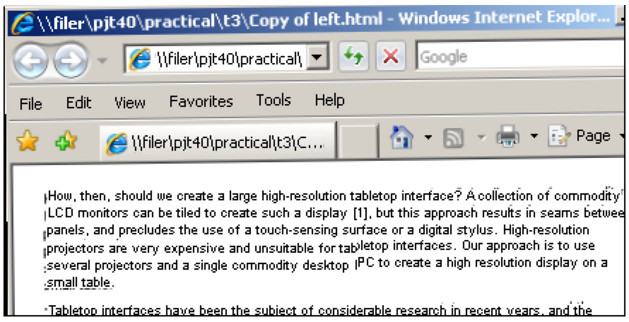
I have reproduced their technique. It identifies high-contrast islands by thresholding the contrast image of the source image, and then performs a morphological close operation to merge closely-neighbouring islands. The technique produces good results under a fairly mild transformation (a horizontal scale by factor 1.005). However, a more realistic transformation would correct for a disparity of 10px over an XGA (1024px$\times$768px) display, and this corresponds to a scale by factor 1.013 and a rotation by $0.75°$. Using this more realistic transformation, the reproduction introduces distracting artefacts (Figure 5.5(b)). The artefacts arise because the technique does not reliably identify words or lines of text as individual islands. I have made some ad-hoc changes to the technique in order to more reliably segment individual words as individual islands, by using knowledge about the likely size and spacing of words, and the foreground and background colours. This adapted technique leads to fewer distracting artefacts for the text shown in Figure 5.5(c). However, it is not clear whether text can reliably be warped using this technique in the general case. This is an interesting future area of research, though longer-term improvements in display technology may prompt a move away from multi-projector displays and eliminate the need for warping.


\\filer\pjt40\practical\t3\Copy of left.html - Windows Internet Explor...
\\filer\pjt40\practical\
Google
File
Edit
View
Favorites
Tools
Help
\\filer\pjt40\practical\t3\C...
Page

How, then, should we create a large high-resolution tabletop interface? A collection of commodity
LCD monitors can be tiled to create such a display [1], but this approach results in seams betwee
panels, and precludes the use of a touch-sensing surface or a digital stylus. High-resolution
projectors are very expensive and unsuitable for tabletop interfaces. Our approach is to use
several projectors and a single commodity desktop PC to create a high resolution display on a
small table.

Tabletop interfaces have been the subject of considerable research in recent years, and the

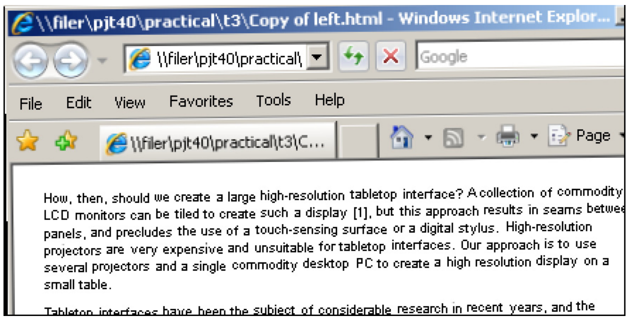
(a)


\\filer\pjt40\practical\t3\Copy of left.html - Windows Internet Explor...
\\filer\pjt40\practical\
Google
File
Edit
View
Favorites
Tools
Help
\\filer\pjt40\practical\t3\C...
Page

How, then, should we create a large high-resolution tabletop interface? A collection of commodity
LCD monitors can be tiled to create such a display [1], but this approach results in seams betwee
panels, and precludes the use of a touch-sensing surface or a digital stylus. High-resolution
projectors are very expensive and unsuitable for tabletop interfaces. Our approach is to use
several projectors and a single commodity desktop PC to create a high resolution display on a
small table.

Tabletop interfaces have been the subject of considerable research in recent years, and the

(b)

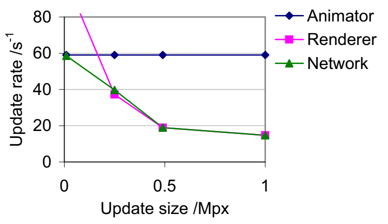

\\filer\pjt40\practical\t3\Copy of left.html - Windows Internet Explor...
\\filer\pjt40\practical\
Google
File
Edit
View
Favorites
Tools
Help
\\filer\pjt40\practical\t3\C...
Page

How, then, should we create a large high-resolution tabletop interface? A collection of commodity
LCD monitors can be tiled to create such a display [1], but this approach results in seams betwee
panels, and precludes the use of a touch-sensing surface or a digital stylus. High-resolution
projectors are very expensive and unsuitable for tabletop interfaces. Our approach is to use
several projectors and a single commodity desktop PC to create a high resolution display on a
small table.

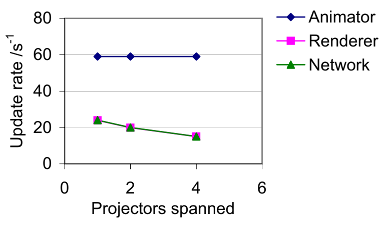
Tabletop interfaces have been the subject of considerable research in recent years, and the

(c)

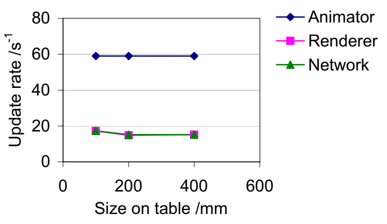
Figure 5.5: Simulation of different warping techniques: (a) bilinear interpolation degrades high contrast features; (b) Hereld and Stevens' technique produces crisp text but with distracting artefacts; (c) the adapted technique produces crisp text without artefacts.



(a)



(b)



(c)

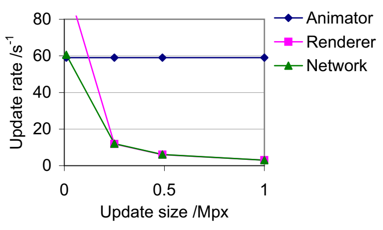
Figure 5.6: Performance using animated tile painting.

5.3 Server, Client, and Protocol Performance

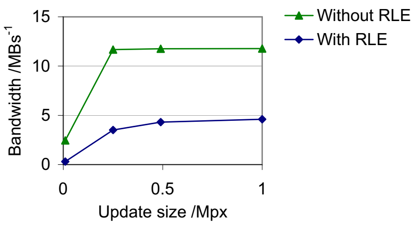
I measured the performance of the server, distribution architecture, and client using four tests. The tests either transformed tiles or repainted tiles, and operations were either driven by an animator thread, or by stylus input. Each test was written using the T3 API. I measured the rate of framebuffer updates over the network received at the client, the rate of the rendering thread, and, for animated tests, the rate of framebuffer updates made by the animator thread. In interactive tests I also measured the approximate latency between moving the stylus and seeing the visual feedback. The stylus was moved at approximately constant speed while observing the approximate distance between the stylus and the visual feedback. The latency was then calculated accordingly.

5.3.1 Animated Tile Painting

A tile was repeatedly repainted by an animator thread rate-limited to 60fps. Tiles of size 100×100px (0.01Mpx), 500×500px (0.25Mpx), 700×700px (0.49Mpx), and 1000×1000px (1Mpx) were tested in sequence. In all cases the tile was sized to 400mm×400mm and positioned to span four projectors. The tile contained an animated square and text at regular intervals.



(a)



(b)

Figure 5.7: Effects of RLE, using animated tile painting: (a) performance without RLE; and (b) network bandwidth with and without RLE.

Figure 5.6(a) shows the results. The animator thread at the server ran at a constant 58fps. At 0.01Mpx, the client received updates at this rate. The renderer thread ran at an even greater frame rate, indicating spare rendering capacity. In this figure, and in other similar figures in this chapter, the rendering frame rate itself is largely meaningless at such points because the renderer would “spin”, counting frames in which no updates were received and no rendering work was done. Such data points are presented merely to indicate that the renderer had spare capacity and therefore did not limit the system. As tile size increased, the renderer reached capacity. The render rate consequently decreased, and the rate of network updates adapted to match. Performance was not limited by the network bandwidth.

I conducted three further animated tile painting tests. The first measured the effects of a tile spanning different numbers of projectors. A 1Mpx tile was sized to 200mm$\times$200mm and positioned to span 1, 2 and 4 projectors. Figure 5.6(b) shows the results. As the number of projectors increased, the render rate decreased, and the rate of network updates adapted to match. Performance was not limited by the network bandwidth.

The second additional test measured the effects of a tile occupying differently sized areas on the display. A 1Mpx tile was positioned to span 4 projectors and sized to occupy 100mm$\times$100mm, 200mm$\times$200mm, and 400mm$\times$400mm. Figure 5.6(c) shows that size had little effect on performance.

Lastly, Figures 5.7(a) and 5.7(b) show the effects of removing run-length encoding compression from the tile update messages. As tile size increased, the rate of data transfer between client and server became the bottleneck and performance decreased rapidly. The animator rate remained at 58fps throughout.

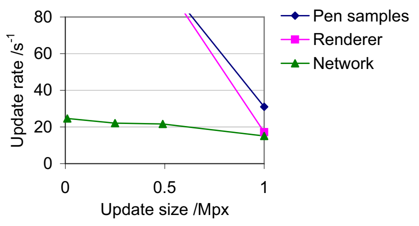


Figure 5.8: Performance using interactive tile painting.

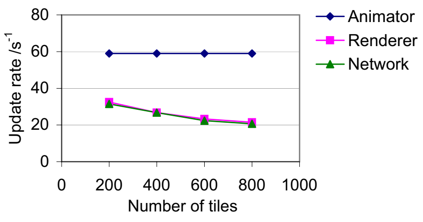


Figure 5.9: Performance using animated tile transformations.

5.3.2 Interactive Tile Painting

A tile was repeatedly repainted in response to stylus movement. Tiles of size 0.01Mpx, 0.25Mpx, 0.49Mpx, and 1Mpx were tested in sequence. In each case the tile was sized to 400mm$\times$400mm and positioned to span four projectors. The tile contained a square that moved to follow the stylus, and text at regular intervals. Figure 5.8 shows the results. At small sizes, the render rate exceeded the rate of network updates, indicating that the renderer had spare capacity and did not limit the system. Again, the render rate itself is largely meaningless at such points, and the axis is scaled accordingly. At 1Mpx the render rate decreased and the rate of network updates adapted to match it. Performance was not limited by network bandwidth. Latency was approximately 0.1s in all cases except for the 1Mpx tile, for which it rose to approximately 0.15s.

5.3.3 Animated Tile Transformations

The display was filled with large numbers of small tiles that rotated and followed predetermined paths using an animator thread rate-limited to 60fps. 200, 400, 600 and 800 tiles of size 100$\times$100px each occupied 40mm$\times$40mm. Figure 5.9 shows

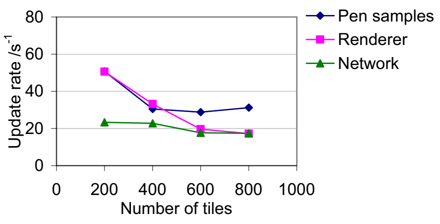


Figure 5.10: Performance using interactive tile transformations.

the results. The animator thread ran at a constant 58fps. As the number of tiles increased, the render rate decreased and the rate of network updates adapted to match. Performance was not limited by the network bandwidth.

5.3.4 Interactive Tile Transformations

The display was filled with tiles that moved to follow the stylus. All other details were the same as the animated tile transformation test. Figure 5.10 shows the results. The rate of network updates remained relatively constant, and was not limited by rendering. At 600 tiles the render rate then decreased, and the rate of network updates adapted to match. Latency was approximately 0.1s in all cases.

5.3.5 Discussion

This investigation of performance tested interactive and animated variants of two operations commonly carried out in tabletop applications.

In tile transformation tests, T3 performed well enough to be useful as a platform, both for the numbers of tiles commonly displayed in tabletop applications and also at the extremes of what application designers might wish to accomplish. Figure 5.11 shows that the animated tile transformation performance also compares favourably to the performance of other systems measured using quite similar tests. These tests were conducted by Miede (2006), and compared the Buffer Framework (Isenberg et al., 2006) (Section 3.3.3), and a more basic implementation that had not been intended for use on a multi-projector display. The tests animated varying numbers of tiles with a side length between 80px and 120px and were conducted using a 5.2Mpx tabletop display composed of four 1280×1024px

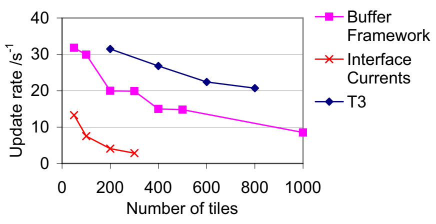


Figure 5.11: Comparison of the effective frame rate of different systems using animated tile transformations. (The effective frame rate for T3 is the network update rate.)

projectors. The display was in both cases driven by a single Matrox QID Pro four-head graphics adapter with vertical synchronisation enabled and a 1.4GHz Intel Xeon processor.

The tile painting tests continuously repainted a tile. This is perhaps less common in tabletop applications. Continuous visual feedback on tabletop interfaces tends to involve transformation of task artefacts and is therefore accomplished using tile transformation. Tile painting is more likely as single infrequent operations, for instance when opening a menu or a new web page or selecting a column of a spreadsheet. In such cases the most important performance metric is the latency between the stylus action and the visual feedback. This was measured in the interactive tile painting test and was sufficiently small that T3 can be considered useful as a platform when updating up to at least 1Mpx and spanning four projectors.

There may nevertheless be a need for continuous visual updates that cannot be abstracted as tile transformations and therefore require repeated tile repainting. The tests identified that the frame rate in such cases depends on the size of the update, the number of projectors spanned by the updated tile, and whether run-length encoding can be applied to reduce the data rate. Run-length encoding compresses blocks of continuous colour and so can be sensibly applied to the majority of conventional user interfaces. In such cases, T3 exhibits good performance for animated updates of 0.49Mpx, and remains usable at 1Mpx when spanning four projectors. Repeatedly updating all 4.7Mpx is likely to result in an unusable frame rate, and I acknowledged this in Section 4.2 when designing the architecture.

Run-length encoding cannot be applied in cases such as animating a shaded three-dimensional scene, or playing video in a tabletop video-editing application. In such cases, the performance is usable at animated updates of 0.01Mpx. However, as the update size increases, the system quickly saturates the available network bandwidth and performance consequently declines severely to 10fps at 0.25Mpx, and continues to decline beyond this. Alternative abstractions might address this by, for instance, broadcasting a stream of compressed video or three-dimensional render primitives to each client.

Overall the results show that performance is sufficient to support a range of high-resolution or remote tabletop applications. They highlight the performance bottlenecks identified in Section 3.3, of large updates, multiple graphics outputs and limited network bandwidth, and illustrate how appropriate abstractions can overcome these limitations in practice. The cases in which the abstractions cannot be exploited expose the performance limitations inherent in the use of commodity networking and graphics hardware to implement high-resolution remote tabletop interfaces.

Improvements in graphics, processing and networking technology are likely to change the performance bottlenecks in the long term. Nevertheless, the bottlenecks reveal two areas for potential short term future improvements.

Firstly, the performance when repainting large tiles was limited by the client's render thread in both tests. This thread reads data from the network socket, decompresses it, copies it to texture memory, and rerenders each graphics output. Performance relies on minimising the allocation and copying of the large image buffers. The client is currently optimised accordingly, subject to architectural constraints. Further performance gains could be achieved by changing the protocol to avoid Java serialisation, by switching to an alternative platform, such as C++, that supports the reading of buffers without type constraints, and by using direct memory sharing instead of TCP sockets when client and server run on the same PC.

Secondly, in some interactive tests, the network update rate was limited to about 25fps. This was not caused by the time taken to process an input event at the server. Input events were sent to the server at a much higher rate, independently of the server load, and large queues and delays would quickly have built up if the server had not processed them at this higher rate. It seems likely that the update rate was instead limited by the adapted client-pull network protocol. Input messages and update request messages from the client are processed by the same sever thread. This may result in longer delays between clients requesting and receiving updates, and consequently lower network update rates. Performance may therefore be improved using alternative adaptive-rate mechanisms.

Further optimisations are also possible in the distribution architecture. Time-sensitive, loss-tolerant messages such as cursor updates could be sent using a UDP-based protocol to avoid wasteful retransmission of lost packets. Switching from Java serialization to a customised byte-stream protocol could result in

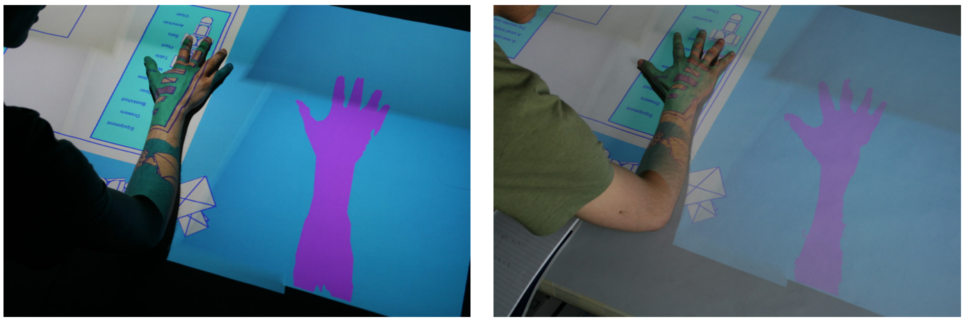

(a) (b)

Figure 5.12: Arm segmentation and representation in (a) dark and (b) light conditions.

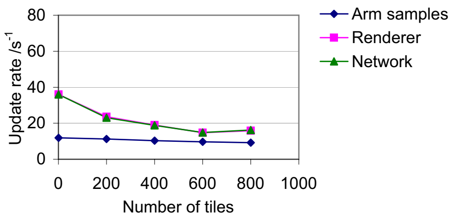


Figure 5.13: Performance using arm segmentation and interactive tile transformations.

smaller message sizes (Gutwin et al., 2006). The adapted client-pull scheme relies on low delays between the client and the server, and so explicitly calculating network capacity and rendering capacity would lead to higher throughput on high-delay links.

5.4 Arm Segmentation and Representation

The arm segmentation and remote representation method is intended not as an input device for controlling the interface but rather as a communication medium for conveying gesture and consequential communication (Section 3.1). Accordingly, the method is evaluated primarily through observations of its use in a collaborative task (described in the following chapter). Nevertheless, several technical aspects are worthy of discussion.

Figure 5.12 shows that the algorithm successfully segments the arm image under different lighting conditions, and that the remote representation faithfully reproduces the shape of the underlying arm. I investigated the performance of the segmentation system using the same apparatus as the previous performance tests and using the interactive tile movement tests. Figure 5.13 shows the results. The computer vision module captured and processed frames at 640px$\times$480px resolution at 15fps which, tests indicate, is the maximum capture rate that can be achieved using the camera. The T3 Java client, however, sent these updates to the server, received updates from the server and rendered the representation all only at 12fps. This remained relatively constant as the number of tiles increased, indicating that the Java client initially had spare processing capacity. This suggests that the apparent bottleneck is the inter-process communication between the natively-compiled computer vision module and the T3 Java client. Nevertheless, the following chapter will show that this performance is sufficient to convey consequential communication and a variety of gestures. The latency between moving the hand and seeing the visual feedback, measured using the procedure in Section 5.3, was approximately 0.1s.

The linear transformation between the camera coordinates and surface coordinates effectively assumes pinhole optics in the camera. In practice, this assumption does not hold, but inaccuracies are small compared to the size of the fingers, hands and arms.

The camera positioning was difficult. The web-cam used in display A, like many web-cams, has a relatively narrow field of view and so must be mounted some distance from the surface to capture the entire workspace. However, the projectors are mounted relatively close to the surface and obscure the view. Mounting the camera obliquely avoids this but introduces further problems. Participants must be seated appropriately to ensure that their bodies do not block the camera view when they lean over the table. The linear transformation compensates for the oblique camera view, but some distortions occur because of the three-dimensional nature of the hand. The fingers of a spread hand, for instance, begin to merge when viewed from an oblique angle. A wider-angle lens may avoid the need for oblique mounting but may introduce greater non-linearities.

5.5 Reuse of Existing User Interface Components

Figures 5.14 and 5.15 provide a short illustration of the reuse of Java Swing user interface components. A short program creates a portfolio and fills it with a third-party Swing JSpreadsheet component. The spreadsheet appears as a legible 250mm$\times$250mm tile and can be passed between collaborators using the

```
public static void main(String[] a) throws Exception {

// Create a new server, d.
PortfolioServer d = new PortfolioServer(
     new ServerSocket(2000),
     new ServerSocket(2001),
     false
);

// create a new portfolio of 600*600 pixels on d
SwingFramePortfolio p = new SwingFramePortfolio(
     d, d.rootPortfolio, new RotateNTranslate(),
     600, 600,
     0, 0
);

// add a JSpreadsheet to the portfolio
p.getFrame().add( new JSpreadsheet(80, 80) );

// make it appear a physical size 250mm*250mm
p.setTileWidthAndHeightInPORT(250.0, 250.0);

// position the tile 300mm inset into the surface,
// rotate it by 0.17 radians  and set scale factor 1.00.
p.setPORTtoPPORT(300.0, 300.0, 0.17, 1.00);

// make it visible
p.setVisibleWhenParentVisible(true);
}
```

Figure 5.14: Source code to creating an interactive tabletop spreadsheet.

Rotate'N'Translate interaction technique (Section 2.2.2) (Kruger et al., 2005). Columns and rows can be selected using the stylus, and formulae typed into the cells. Multiple collaborators can simultaneously work in different spreadsheets. Remote collaboration is possible by connecting a second client to the server, and does not require modification of the program.

This short example provides a starting point for designing collaborative tabletop data analysis interfaces. Potential areas of investigation include: awareness visualisations for dependencies between multiple spreadsheets and graphical representations; management of large datasets and analyses using the tabletop storage mechanisms, such as piling, reviewed in Section 2.2.3; and collaborative debugging of multiple interdependent spreadsheets. I describe further examples of combining existing user interface components and tabletop interaction techniques in Section 5.6.

I measured the effects of Swing on performance using variants of the animated-

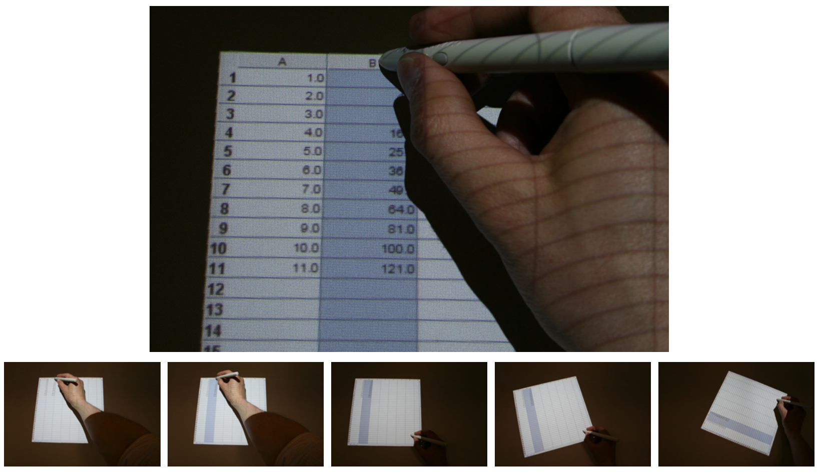


Figure 5.15: An interactive tabletop spreadsheet that can be edited and moved on the table, created using the source code in Figure 5.14.

and the interactive tile painting tests, and using the same equipment as before. In each case, the tile corresponded to a Swing window and used Swing procedures to paint and to trigger repainting. The interactive variant received user input using the Swing event system. Figures 5.16(a) and 5.16(b) show that using Swing had no negative effects on performance in either case. The latency was approximately 0.1s for all cases except for the 1Mpx tile, for which it rose to approximately 0.15s, similar to the non-Swing tests.

The ability to reuse Swing components in tabletop applications is not without

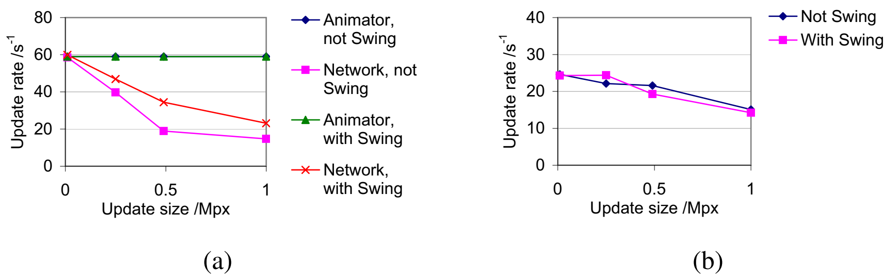


Figure 5.16: Effects of Swing on performance, using (a) animated and (b) interactive tile painting. The update rate for interactive tile painting is the network update rate.

limitations. In practice, these mirror the problems outlined in Section 4.7:

Multi-user input. Swing components are designed only for a single user. In the spreadsheet application, for example, it is not possible for two users to concurrently enter text or select rows and columns within the same spreadsheet. Concurrent interaction is, however, possible, provided collaborators are working in different portfolios. Similarly, Swing maintains only a single "clipboard" for cutting and pasting items, which must be shared between collaborators using social protocol.

Bimanual or gestural input is not processed directly by existing Swing components, which assume a single three-button mouse. However, T3 uses extended Java Swing event objects that also contain the originating T3 event object, which in turn can contain such information. Customised Swing components can therefore access this information and respond accordingly to bimanual or gestural input.

Input lifting violates the Java event model and causes Swing applications to see the mouse jump between disparate locations. Mouse enter and exit events are nevertheless generated correctly and, in practice, the jumps have not caused observable problems in any of the applications.

These limitations may change over time as windowing systems like Microsoft Windows, and their standard component libraries, begin to support multi-touch interfaces.

A further limitation arises because of the mechanism for capturing the raster output of rendering these user interface components. Popup entities, such as menus and list boxes, are transparently repositioned by Swing to fit on the screen at the server. This causes problems when capturing their render output and, in practice, cannot be used reliably with this method. Components that render video or animated three-dimensional scenes tend to achieve performance by interfacing directly with the graphics adapter and so their raster output is not easily captured using this method.

5.6 Further Projects

The technical evaluation has so far focussed on constrained tests and particular sub-methods. I now describe a variety of prototype tabletop applications implemented using T3, in order to demonstrate that it enables researchers to investigate novel interfaces that could potentially benefit real-world tasks.

Though informed by studies of real-world tasks, the presented works are nevertheless prototypes and are unlikely themselves to be useful in a real-world context because they lack necessary features such as saving of state, and integration with other tools. I have also not conducted field evaluations. The practical barriers to deploying and establishing a prototype technology such as a multi-projector display in a field study are considerable. Some of the applications (such as web-browsing) are further unlikely to be adopted until large high-resolution displays become ubiquitous.

As discussed in Chapter 3, however, large displays are becoming more affordable and more widespread: witness the large flat-panel displays as advertising boards in high-street shops, and Microsoft's recent Surface project to retail tabletop systems. As this trend continues, field studies, and ultimately consumer products, will become feasible. The design of such interfaces can be informed by the laboratory prototypes that researchers can construct at present. In this section then, I aim to show that T3 is useful in aiding such researchers and can be applied to create novel prototypes to address a variety of real-world tasks. Some of the applications were constructed by me in order to investigate T3, while others were implemented by final-year undergraduate students under my supervision. An additional project is being undertaken by Mark Ashdown and others at MIT.

5.6.1 Web Browsing

The first prototype investigates tabletop web-browsing, primarily among co-located collaborators. Morris (2008) conducted a survey of 204 knowledge workers to investigate collaborative web search practices and needs. 88% reported having watched over someone's shoulder and suggested alternative query terms, while 85% reported having shown a personal display to others to share search results, and 30% reported having instant-messaged search results to others. Common themes in the collaborative tasks people reported included travel planning, online shopping, literature searches, social planning (for example, to choose restaurants and films), and real-estate searches. Though respondents reported engaging in such activity relatively infrequently (only 14% reported that they did so once each week), the survey also established that respondents were frustrated by the inadequate support for collaborative search in current tools. This suggests that collaborative search may become a more frequent activity given more appropriate tools.

Two common themes in the reported obstacles to collaborative search were the difficulty in working independently while maintaining awareness of each other's activities, and a difficulty establishing a shared focus with remote collaborators. Amershi and Morris (2008) in a further study conducted interviews with seven people who worked in areas where co-located collaborative search was a common activity: three teachers; two librarians; and two technology researchers in developing regions. They established several further limitations of collaborative search technology. Collaborators have difficulty: contributing and learning, because of the single input device; maintaining awareness of each other's suggestions; adapting to each other's reading speeds; pointing to the display in larger groups; working independently; and keeping track of findings.

Researchers have developed interfaces to address the problems of collaborative web search using several approaches: a conventional groupware approach for remote collaborators (e.g. Greenberg and Roseman, 1996; Morris and Horvitz, 2007); multiple-PDAs or laptops for co-located collaborators (e.g. Han et al., 2000; Paek et al., 2004); and a single desktop PC with multiple mice (e.g. Amershi and Morris, 2008). A tabletop approach to web search, in which pages are presented as task artefacts that can be moved and grouped into containers, has not previously been postulated but may similarly address several of the frustrations (such as ability to work independently and maintain awareness) . Such an approach might additionally support the existing beneficial work practices reviewed in Chapter 2, such as fluid transitioning and territorial coordination. Though at present it seems unlikely that collaborators would obtain and use a large display for the sole purpose of collaborative web-browsing, this may change as large displays and collaborative interfaces become more affordable and prevalent in homes and workplaces.

Such a project presents many potential areas of investigation. The affordances of horizontal displays for reading differ greatly from those of paper task artefacts such as newspapers or brochures, and of conventional desktop computers and laptops (Morris et al., 2007). It is unclear whether the more cumbersome ergonomics of reading on a horizontal surface are offset by the potentially improved social experience and display real estate of tabletop interfaces. Large pages must be presented in such a way that they can be shared between collaborators. The extra display real estate enables investigation of novel visualisations of large numbers of linked pages. Finally, early experiences may lead to tailored tabletop applications such as a "Wikipedia table" or a "Facebook table".

Using T3, I created a tabletop web-browsing application that allows collaborators to open new web pages by following links, and to pass them between themselves (Figure 5.17). The high-resolution capability enables web pages to appear legible yet sufficiently small that multiple collaborators can each read their own pages and pass them around. This basic application was created by reusing a third-party Java Swing web browser component, combined with the Rotate'N'Translate interaction technique (Section 2.2.2) (Kruger et al., 2005). The implementation required about 60 lines of Java source code written in about 1 hour.

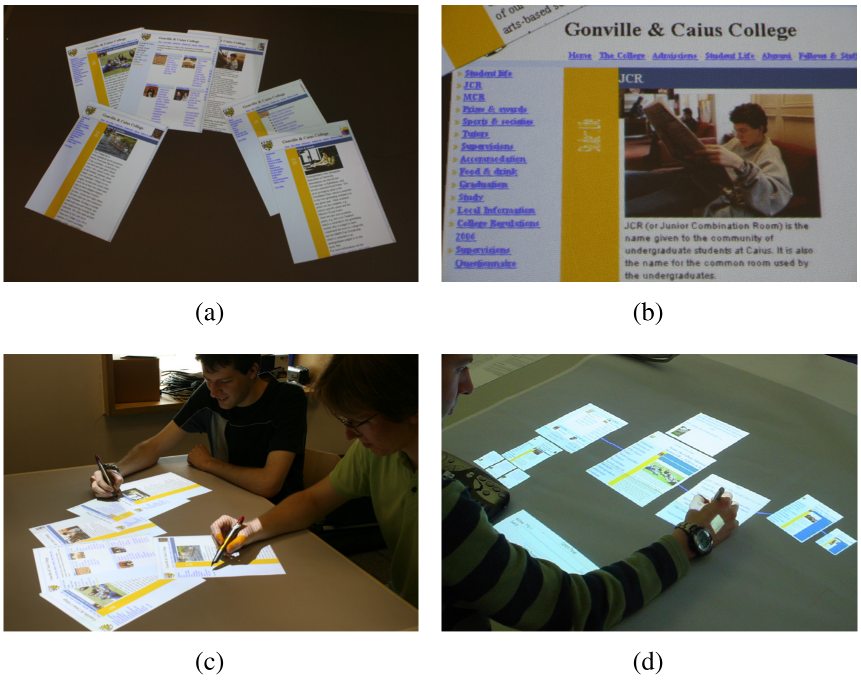


Figure 5.17: Collaborative web-browsing ((d) taken by Richard Russell).

Richard Russell, an undergraduate student, has enhanced this basic functionality and begun to explore novel visualisations by creating a personal history tree for each user. The tree illustrates the order in which pages have been visited by displaying linked expandable page thumbnails on the display (Figure 5.17(d)). Page expansion and shrinking is accomplished using tile scale factor, and links between portfolios form the edges of the tree.

Further exploration of this area is beyond the scope of this dissertation, but the project highlights the potential of the T3 to enable investigations of tabletop interfaces that were previously considered infeasible in the scope of a research project.

5.6.2 Remote Document Collaboration

This prototype investigates document review meetings, in which collaborators discuss and collaboratively annotate draft documents that they have each brought to the meeting. These meetings are common in knowledge-worker organisations in which teams regularly integrate information from different documents into a single report.

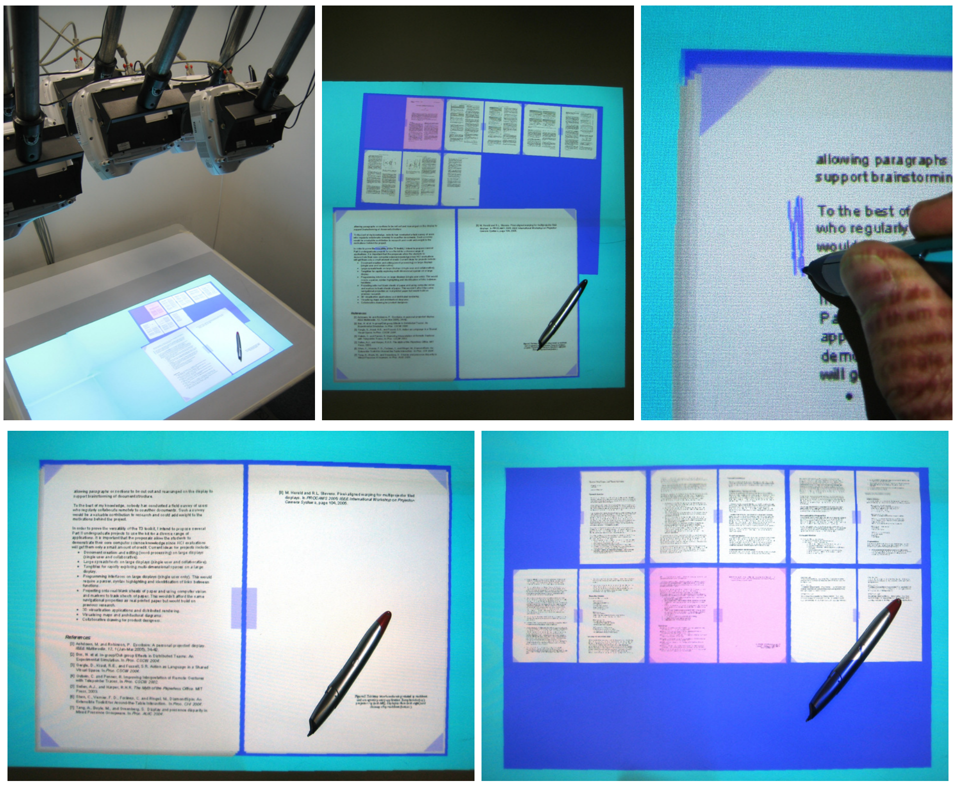


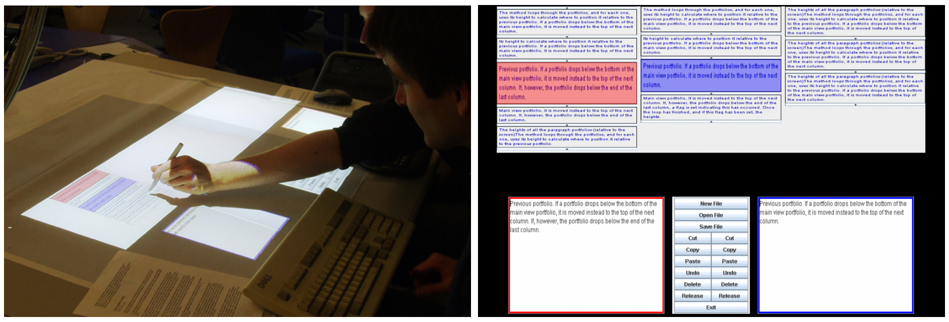
Figure 5.18: Text document interface for remote document collaboration.

Sellen and Harper (2003) investigated the use of paper in such meetings in an ethnographic study of the International Monetary Fund (IMF). The IMF at the time was technology-rich and consisted of about 900 professional economists who produced about 4500 reports during the six-month period of study. Most of the reports were co-authored, and consequently the economists spent time editing and reviewing each other's work to ensure that the constituent sections and figures, produced by different authors, integrated to produce a coherent whole. In a 5-day diary study of 8 economists, Sellen and Harper found that 44% of the time spent revising documents took place in face-to-face meetings. During the study almost 2750 minutes were reportedly spent in such meetings in total; on average just over 1 hour per day per economist. Despite the available technology, 82% of time in these meetings involved paper documents alone, rather than a mix of electronic and paper documents or electronic documents alone. They describe how the paper documents in such meetings formed a focus for discussion without distracting, were easily navigated to show the relevant pages, could be placed to compare pages side-by-side, and were easily annotated. They observe that the time spent by the economists in different activities is similar to surveys of other knowledge-worker organisations, and further state that although they have since examined "many other workplaces: hospitals, telecommunications companies, manufacturers, design companies, and many more", they "have yet to see a place where paper does not figure in the same kinds of activities". I speculate that tutoring meetings, in which collaborators review homework, or annotated lecture materials, are likely to involve similar activities.

Such activities are difficult to conduct when collaborators are remote. Commercially-available shared-workspace technology (such as a word processor in a Microsoft Netmeeting session) suffers awareness problems and does not typically allow collaborators to view different areas of the document concurrently. Furthermore, screen-based text documents suffer the problems highlighted by O'Hara et al. (2002) and O'Hara and Sellen (1997). They showed that navigating and annotating was more disruptive on the screen than on paper, and that readers of screen-based documents had more difficulty determining their location in the document. Additionally, this approach does not support easy switching between documents or side-by-side comparison. Video-conference systems use document cameras to provide a remote-tangible approach, similar to the Double DigitalDesk (Wellner, 1993a,b; Wellner and Freeman, 1993). As discussed in Section 2.7.3, this allows both collaborators to see the document, but only one can annotate, turn pages or otherwise manipulate.

High-resolution remote tabletop interfaces offer an alternative approach to this problem: multiple documents can be viewed concurrently by remote collaborators; stylus-based annotation is likely to be richer and less disruptive than conventional screen-based annotation using a keyboard; and the larger display size, tabletop interaction mechanisms and stylus input may allow more effective document navigation mechanisms. The Escritoire (Ashdown, 2004; Ashdown and Robinson, 2005) exhibited promising early results using this approach, but there has been no investigation of visualisations for multi-page documents, or field study of remote tabletop interfaces for such tasks.

Again, further exploration of this area is outside the scope of this dissertation, but the project serves to illustrate the potential of T3 to enable further research. As a first step towards investigating such interfaces, I have used T3 to create an interface in which multi-page text documents appear as appear as "virtual sheets of paper" in the shared tabletop workspace (Figure 5.18). Participants can pass documents to each other using a stylus, browse within a document using a thumbnail view, and annotate pages using the stylus. The high-resolution display enables small text to be projected legibly at size 12pt. Multi-page documents appear showing two sheets at once, rather like an open book. The application was implemented using T3 in about three days.



The method loops through the portfolios, and for each one, uses its height to calculate where to position it relative to the previous portfolio. If a portfolio drops below the bottom of the main view portfolio, it is moved instead to the top of the next column.

Its height to calculate where to position it relative to the previous portfolio. If a portfolio drops below the bottom of the main view portfolio, it is moved instead to the top of the next column.

Previous portfolio. If a portfolio drops below the bottom of the main view portfolio, it is moved instead to the top of the next column. If, however, the portfolio drops below the end of the last column.

Main view portfolio, it is moved instead to the top of the next column. If, however, the portfolio drops below the end of the last column.

The heights of all the paragraph portfolios (relative to the screen)The method loops through the portfolios, and for each one, uses its height to calculate where to position it relative to the previous portfolio.

The method loops through the portfolios, and for each one, uses its height to calculate where to position it relative to the previous portfolio. If a portfolio drops below the bottom of the main view portfolio, it is moved instead to the top of the next column.

Its height to calculate where to position it relative to the previous portfolio. If a portfolio drops below the bottom of the main view portfolio, it is moved instead to the top of the next column.

Previous portfolio. If a portfolio drops below the bottom of the main view portfolio, it is moved instead to the top of the next column.

Main view portfolio, it is moved instead to the top of the next column. If, however, the portfolio drops below the end of the last column, a flag is set indicating this has occurred. Once the loop has finished, and if this flag has been set, the heights.

The heights of all the paragraph portfolios (relative to the screen)The method loops through the portfolios, and for each one, uses its height to calculate where to position it relative to the previous portfolio. If a portfolio drops below the bottom of the main view portfolio, it is moved instead to the top of the next column.

The heights of all the paragraph portfolios (relative to the screen)The method loops through the portfolios, and for each one, uses its height to calculate where to position it relative to the previous portfolio. If a portfolio drops below the bottom of the main view portfolio, it is moved instead to the top of the next column.

The heights of all the paragraph portfolios (relative to the screen)The method loops through the portfolios, and for each one, uses its height to calculate where to position it relative to the previous portfolio. If a portfolio drops below the bottom of the main view portfolio, it is moved instead to the top of the next column.

Previous portfolio. If a portfolio drops below the bottom of the main view portfolio, it is moved instead to the top of the next column. If, however, the portfolio drops below the end of the last column.

Previous portfolio. If a portfolio drops below the bottom of the main view portfolio, it is moved instead to the top of the next column.

Figure 5.19: Collaborative text editing (photo taken by Tom Matthews).

5.6.3 Large-format Programming Interfaces

The third prototype uses the high-resolution capabilities to investigate large collaborative programming interfaces. This work is motivated by two problems of conventional programming interfaces. Firstly, writing and debugging computer programs typically requires an understanding of dependencies between components of the program in different parts of the same source file, or different source files entirely. On a conventional desktop computer with limited screen size, this typically requires the programmer to navigate regularly between different parts of source code. Researchers have recently shown that visualisation techniques such as fish-eye lenses integrated into the development environment go some way to addressing this problem (e.g. Jakobsen and Hornbaek, 2006). Tabletop interfaces offer a much larger form-factor but there has been little investigation of tabletop programming interfaces. We do not know then whether the large size offers benefits to programmers and, if so, what visualisations are most appropriate.

Secondly, the software engineering methodology of Extreme Programming (Beck, 1999) encourages programmers to work closely-coupled in pairs at a single computer, sliding a keyboard and mouse back and forth. By explaining design decisions to each other as they work, Beck suggests that they narrow the design-space more quickly when solving problems and produce more appropriate source-code with fewer bugs. Williams et al. (2000) examine the case for this pair programming practice, reviewing controlled experiments and organisational surveys. They conclude that, though often rejected as wasteful, pairs working together achieve high-quality source code in less time than it would have taken two individuals working separately to achieve the same quality. However, pair-programmers continue to use software development tools intended for a single user and, although interfaces designed for collaborative programming might offer even better results,

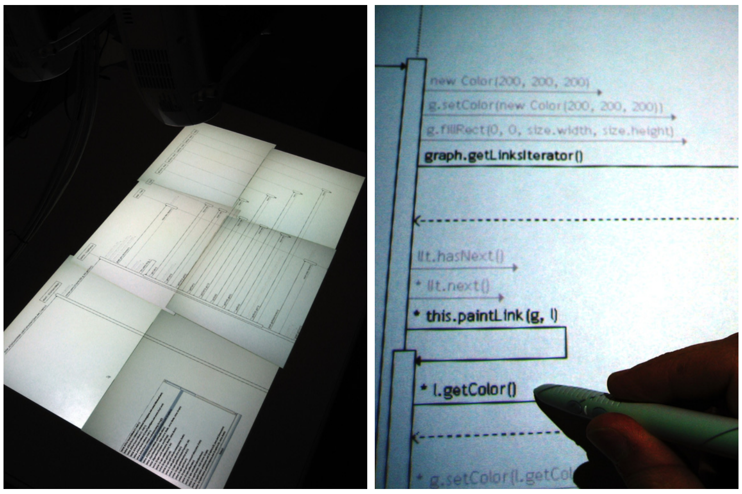


Figure 5.20: Interactive UML sequence diagrams.

there has been little investigation in this area. Tabletop interfaces are of particular interest because they can enable collaborators to work independently while maintaining awareness of each others' actions, offer space for collaborators to work side-by-side as well as novel visualisations (as I propose above), and can additionally support the existing beneficial work practices reviewed in Chapter 2, such as fluid transitioning and territorial coordination.

Two undergraduate students used T3 to create two applications as a starting point to investigate such interfaces. The first application is a collaborative text editor created by Tom Matthews. The text document is presented in columns on the display surface (Figure 5.19). Collaborators sit side-by-side, each with their own keyboard. Each collaborator proceeds by selecting a paragraph of text from the columns using the stylus. This paragraph then appears shaded to indicate that it is currently being edited. The text appears in a small box immediately in front of the collaborator, along with a flashing cursor, and the collaborator can enter text and otherwise manipulate the paragraph using the keyboard. The paragraph in the column updates to reflect the changes. The text entry box is implemented using a Swing text entry component, and each paragraph is implemented as a portfolio.

The second application, created by Rowan Hill, presents interactive UML sequence diagrams as a novel interface for debugging programs. The diagrams fill the display and are annotated with the underlying source code (Figure 5.20). Arrows indicate method calls and can be expanded and collapsed by pressing with the stylus. Dragging the stylus causes the entire diagram to move on the display,

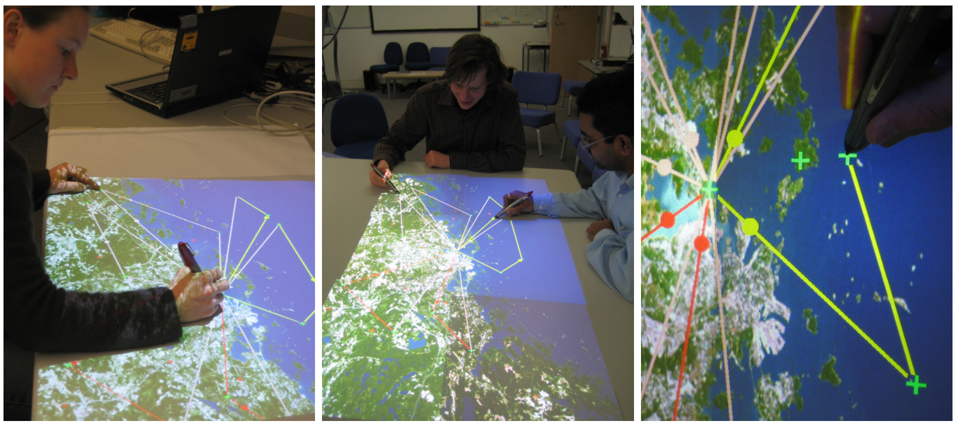


Figure 5.21: Remote time-sensitive targeting without remote arm representations.

rather like a tablecloth. The interface is implemented in T3 as an array of tiles, each forming a portion of the diagram. Dragging causes all the tiles to move, and tiles that leave the display are repositioned to enter the display at the other side, showing a different portion of the diagram. Expanding a method call causes all the tiles to be repainted.

5.6.4 Time-sensitive Targeting

The fourth prototype investigates co-located and remote interfaces for map-based collaborative command-and-control tasks. Cummings and Mitchell (2007) describe one such command-and-control task, the supervisory control of unmanned aerial vehicles (UAVs), commonly used in military intelligence, surveillance and reconnaissance missions. Although presently multiple operators are required to control a single UAV, they suggest that technology will improve to automate the lower-levels of control such as the motion control and navigation. This will leave a single operator free to control one, or even many UAVs by merely specifying routes and time-constraints using a map-based interface. Based on this premise, Scott et al. (2007) investigate interfaces to aid teams of such operators using a hypothetical scenario in which a hostile area must be monitored closely using UAVs as a convoy passes through. They identify the awareness requirements of the team supervisor and use these to design an interface using a conventional GUI approach in which each team member sits at a different display. A tabletop approach was not considered but, designed appropriately, might enable greater awareness among team members, while support for work practices such as territoriality might aid hand-off of UAVs between operators. A tabletop interface for such a task is also potentially useful in laboratory investigations comparing remote and co-located collaboration. The experimenter can manipulate the stress level and measure team performance to establish the cost of working remotely.

Stephen Williams, an undergraduate student, has created an interface for a team of co-located or remote tabletop collaborators (Figure 5.21). The interface presents the collaborators with a map on which UAVs are represented as markers and move along polygonal paths between waypoints. Collaborators must edit the paths using their styluses in order to avoid danger areas and to reach targets. As the mission unfolds, further danger areas and targets emerge and must be addressed within a certain time period.

Updates and UAVs are not assigned to any particular collaborator and so collaborators must coordinate their work and maintain an awareness of each other's actions. A comparison of team performance under both co-located and remote conditions might therefore determine empirically the extra effort required to work remotely, and how this is affected by interventions such as seating position and remote arm representations. Further exploration of this area is outside the scope of this dissertation. Nevertheless, the work illustrates how T3 can be used to create such an interface. The map, the waypoints and the markers are implemented as tiles, and the lines of the polygonal path as links. Markers are animated to move along the links using the T3 animator facility. The interface can be used in co-located and remote conditions without modification.

5.6.5 Distributed Emergency Response Coordination

The fifth prototype investigates collaboration within distributed emergency response teams undertaking an urban search and rescue task in which they search damaged buildings and rescue victims. This work is conducted by Mark Ashdown, Missy Cummings and other researchers at MIT. The team consists of field personnel coordinated by leaders in a command centre. All team members rely on map data, but while the leaders can use large paper maps or a large display, field personnel are limited in the physical size of the devices and maps they can carry. In order to promote awareness among team members and to enable sketching and gesturing on the maps, Ashdown and Cummings (2007) propose creating a shared workspace between small handheld computers carried by field personnel and a large collaborative display in the command centre (Figure 5.22). Previous work offers little guidance as to how to create shared workspaces between such asymmetric devices, and the researchers are creating a number of prototypes to

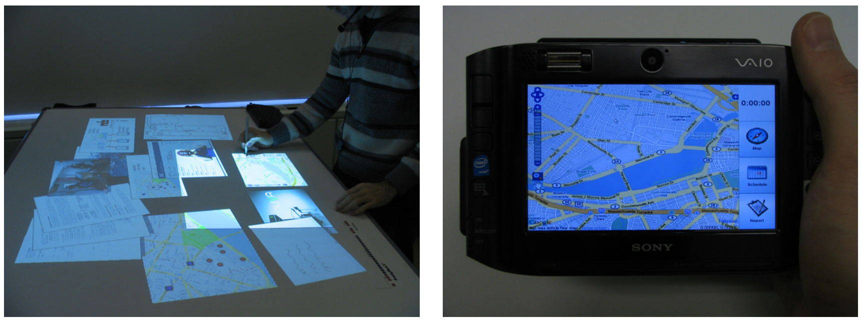

(a) (b)

Figure 5.22: Distributed emergency response coordination: (a) a large display in the tactical command centre is linked to (b) handheld displays for field personnel. (Photos supplied by Mark Ashdown.)

investigate this issue. They have implemented a map-based tabletop interface to support urban search and rescue by running Google Maps in a third-party Java Swing web-browser component in T3. They use the T3 client to provide a shared workspace on the hand-held display, and are currently adapting it to implement a variety of visualisation mechanisms suitable for such asymmetric collaboration.

5.7 Overall Discussion

Having discussed the individual sub-methods, it remains to discuss the limitations of the method as a whole. Although I have shown that the method has been used to implement a wide variety of interfaces, its versatility is inherently limited by the tile abstraction. Section 5.3 has shown how exploiting this abstraction has enabled T3 to achieve a useful level of performance despite the bottlenecks identified in Section 3.3. Interfaces, such as fish-eye lenses, that cannot easily be implemented using the tile abstraction would result in large data transfers and lower performance if implemented using the present T3 architecture. A higher-performance fish-eye system would require additional abstractions at the server and client. The tile abstraction also limits the versatility of the event-routing architecture. T3 currently routes events directly to the tiles on which they occur, and so further work would be required to route, for example, noisy event locations to the nearest interactive element on the table, or to enable gestures such as circling a tile with a pen. Nevertheless, in each of these cases the T3 implementation provides a sound basis from which to work.

Multi-touch surfaces would enable investigation of more expressive gestural input, but were not readily available during the course of this research. There is nevertheless scope to integrate such a device into this input scheme. Each hand would be represented in protocol messages as a different point input device. Each hand would also be represented by a single surface location, such as the centroid of the contact points, to enable routing of events to portfolios. The number of contact points, their contours, and information about hand gestures, could be represented in protocol messages using the additional available fields.

The method is limited to a single client computer controlling each display, which in turn limits the display size. Current motherboards support up to three graphics adapters, each with up to two heads. Current projector prices rise rapidly beyond SXGA+ resolution (1400px$\times$1050px) and therefore the maximum display size is currently 8.8Mpx. At 2.5px/mm this results in an area of approximately 1.4m^2. Larger displays with high spatial resolution require a cluster of PCs for distributed rendering, and specialised software architectures, as described in Section 3.3.

Shortly after the technical work was completed, other researchers reported two similar systems designed to support remote collaboration and tabletop interaction. Firstly C-Slate (Izadi et al., 2007) uses a digitising tablet with an integrated 21" 1.9Mpx LCD display. A novel stereoscopic camera system enables multi-touch input and also captures collaborators' arms and any physical objects on the display, which are then rendered remotely. The authors demonstrate document annotation using the tablet stylus, and rotation, translation and scaling of task artefacts using multi-touch gestures. The system uses a model-synchronising replicated architecture in which mode changes, ink strokes and task artefact transformations are transmitted between the sites. However, the small screen size allows only a single page to be displayed at any time, and precludes co-located collaboration or collaborators working in separate areas of the workspace. Secondly, Digitable (Coldefy and dit Picard, 2007) uses a single front-mounted projector and captures arms to render remotely using a camera and the computer vision method described in Section 4.5. It uses an input-event synchronizing replicated architecture, and the authors demonstrate a virtual jigsaw puzzle application.

5.8 Chapter Summary

In undertaking the technical work, I set out to develop a method and implementation to enable the investigation of the hypotheses developed in Chapter 3. Section 3.3 found that existing systems could not easily be adapted to fulfil such a goal. Furthermore, it established that a new system could be designed to enable wider exploration of remote tabletop interfaces by supporting high-resolution applications using a multi-projector display, and by supporting rapid prototyping through reuse of user interface components. In this chapter I have presented an evaluation of this technical work in order to characterise its limitations and identify areas for future improvement. The early sections evaluated and discussed specific sub-methods: the geometric and photometric correction of the multi-projector display; the reuse of existing user interface components; the arm segmentation and remote representation; and the performance of the server, client and distribution architecture. The tests show that these sub-methods perform sufficiently to be useful as platform from which to create high-resolution or remote tabletop interfaces, and also highlight areas for future technical work. Finally, Section 5.6 presented a range of projects undertaken using T3 by myself and by others. These illustrate the utility and versatility of the method, and show that the combination of rapid prototyping, high-resolution, remote collaboration, and tabletop interfaces, enables rapid exploration of a range of novel collaborative interfaces.

Chapter 6

Awareness and Coordination in Remote Tabletop Collaboration

In the preceding chapters, I have developed design guidelines for remote tabletop interfaces, hypotheses about their use in practice, and methods for their technical construction. This chapter presents an empirical investigation of these interfaces in order to better understand such collaboration and to inform their future design.

The study investigates some of the questions raised in Chapter 3:

Q1. Which of the following mechanisms contribute to workspace awareness?

> **Q1a.** High-fidelity representations of arms, displayed in the workspace, enable rich gesturing for intentional communication.
>
> **Q1b.** Direct manipulation techniques enable task artefact feedthrough.
>
> **Q1c.** High-fidelity representations of arms, displayed in the workspace, together with direct input devices, enable consequential communication when used with local, direct manipulation techniques.

Q2. Does the hypothesised high level of awareness enable collaborators to work in a range of coupling styles, with frequent, fluid transitions between, similar to co-located tabletop collaboration?

Q3. Do remote tabletops support territorial partitioning of space as a coordination mechanism?

Q4. How is territorial partitioning affected by an overlaid or non-overlaid seating arrangement?

These questions investigate both awareness (Q1 and Q2) and coordination (Q3 and Q4) in remote tabletop collaboration. The remaining questions are not addressed directly by this study and relate to the awareness contribution of avoiding workspace navigation (Q1d), and task artefact orientation (Q5 and Q6).

This chapter begins with a discussion of methodology and description of the method and initial observations. I then present results for awareness and coordination, before concluding with a discussion.

6.1 Methodology

HCI has traditionally favoured the quantitative methodology of controlled experiments, in which hypotheses about different conditions are evaluated using quantitative measurements in a laboratory. An effect is identified as a difference in the mean measurements among different conditions, and an analysis of the variance then provides a probability of validity, that the effect did not happen by chance. Controlled experiments emphasise objectivity and reproducibility. They offer an easily-interpreted result when investigating questions such as whether two interfaces differ in task-completion time, or some objective measure of output.

Controlled experiments are, however, limited in investigations of collaborative activity. They rely on control of confounding variables in order to discern effects of the independent variable. In a social setting, the confounding variables are often numerous and difficult to control. Control of confounding variables using actors and scripts, for instance, can also reduce realism and inadvertently change the collaborative activity itself, leading to poor generalisability. This reduced realism is further typically designed with the hypotheses in mind, and so potentially-interesting yet unanticipated effects can be inadvertently rendered unobservable by the strict controls. Lastly, even accepting control of confounding variables and the associated problems, the effects of interest in collaborative activity can sometimes in any case be difficult to measure directly and quantitatively.

HCI, and particularly the CSCW sub-field, have addressed these problems using field studies and by adopting qualitative methodologies of social sciences. Researchers conducting field studies of workplace technology have adopted an ethnographic approach, documenting users' interactions in the context of their social environment (Hughes et al., 1994). The ethnomethodologically-informed variant of ethnography is particularly prominent in CSCW (Shapiro, 1994). It involves extensive analysis of conversation and gesture in a workplace in order to identify the implicit social aspects of work and technology (e.g. Heath and Luff, 1992; Heath et al., 1994). Such field studies enable investigation of the social processes underpinning collaborative work in a highly realistic setting, and investigation of unanticipated phenomena. However, the reliance on objective interpretation by the investigator leads to concerns about bias, and the findings may have limited generalisability beyond the immediate social setting of the investigation. There are also practical barriers to field studies of new technologies such as the difficulty of deploying and establishing prototype technology in an organisation for a period of study.

McGrath (1984a) compares research methodologies on dimensions of precision, realism, and generalisability, and shows that, though all are desirable, increasing one dimension inevitably decreases another. Controlled experiments and field studies maximise precision and realism respectively. Quasi-naturalistic studies, sometimes called experimental simulations, are a compromise between these methodologies. The investigator constructs in a laboratory setting a collaborative task and situation that is designed to make particular effects observable. Participants are introduced to the situation and collaborate naturally for limited time periods so that the effects arise in contexts that the they have themselves created. Compared to controlled experiments, such studies offer greater realism, naturalistic data for qualitative analysis, and greater scope for exploration, but reduced precision. Compared to field studies, they offer fewer practical barriers and greater scope for precision when using quantitative techniques, but reduced realism. These quasi-naturalistic studies have been used widely to evaluate co-located and remote collaboration technologies (e.g. Gaver et al., 1993; Fraser, 2000).

Field studies, quasi-naturalistic studies, and controlled experiments each have strengths and weaknesses, and the choice of methodology depends upon the aims of the research. Remote tabletops are prototypical and would be difficult to establish and study in an organisation. Their novelty reduces the scope for anticipating effects and necessitates an exploratory approach. The awareness hypotheses and work practices under investigation are difficult to measure quantitatively, limiting the scope for a quantitative hypothesis-testing approach, although some specific differences among conditions can be quantified and tested statistically. Accordingly, this study adopts a hybrid approach, similar to other investigations of collaborative technologies (e.g. Kruger et al., 2004; Scott, 2005; Tang, A., et al., 2006b), using a quasi-naturalistic study and drawing on quantitative and qualitative approaches as appropriate.

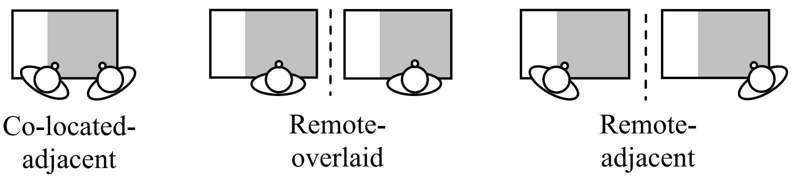


Figure 6.1: Study conditions. Grey indicates interactive areas of the tables.

6.2 Method

6.2.1 Study Design

The study investigated pairs of participants collaborating to complete a design task using both co-located and remote tabletop interfaces. The details of the technology, task, and interaction are described in the following sections.

The study used a within-subjects design with one independent variable consisting of three levels (Figure 6.1):

- In the *co-located-adjacent* (CA) condition, both collaborators in the pair sat in the same room, at the same display, positioned in seats at adjacent corners of the display.
- In the *remote-adjacent* (RA) condition, the collaborators sat in separate rooms at different displays, linked together using the remote tabletop system. The collaborators sat in the same seating arrangement relative to the shared workspace as in the co-located-adjacent condition. Speakerphones provided an audio link between the rooms.
- The *remote-overlaid* (RO) condition was similar to the remote-adjacent condition, except that collaborators sat in the overlaid seating arrangement, both positioned in the same place relative to the shared workspace.

Each pair of subjects tested each of the three conditions. The presentation order of the conditions was counterbalanced using a Latin square. Each pair completed a different design brief in each condition, and the three briefs were always presented in the same order. The briefs are described in the following subsection.

I chose the adjacent seating arrangement over other non-overlaid seating arrangements, such as a corner arrangement or opposite sides. A pilot session comparing a remote corner arrangement with a remote overlaid arrangement had found that participants felt so frustrated in the corner arrangement that they had moved their chairs to the adjacent arrangement. This suggested that the major effect arose from sitting in a corner arrangement versus the same side of the table, rather than because the same-side arrangement was overlaid. Comparing the adjacent and overlaid conditions somewhat controls this confounding variable because participants sit on the same side of the table in both cases.

Data was collected and analysed from five sources:

- Log files recorded the locations and times of each participant's interactions with the system.
- A video record of the participants' interactions enabled coding for quantitative analysis and also a richer, more qualitative analysis of behaviour.
- Questionnaires administered after each condition asked participants about their perceptions using 7-point Likert-scales, alongside open-ended questions about difficulties encountered, what they would like to improve, and how they felt about the seating arrangement.
- A questionnaire administered at the end of the study asked participants their overall preference of remote condition, and their reasons.
- Semi-structured interviews conducted after the study allowed exploration of questionnaire responses.

This study adopts a hybrid approach, using both quantitative and qualitative techniques. Specific dependent variables and analysis techniques are described alongside the results.

6.2.2 Task

The study used a furniture layout task in which participants were asked to work together to arrange diagrammatic furniture on a floor plan to fulfil a design brief. For example, one such brief asked participants to design a new communal space for graduate students and to provide, among other things, as much seating as possible, and areas for reading and serving drinks. The other briefs were similar and asked for designs for a library and for a research lab. Appendix C describes the actual briefs. At the end of the task, participants gave a short joint presentation and answered questions about their solution.

Several factors determined this choice of task. As a generation activity (McGrath, 1984b), it involves manipulation of task artefacts and necessitates construction, organisation and exploration. As such it is similar to the studies of awareness in shared workspaces (e.g. Gutwin and Greenberg, 1999) and the studies of tabletop collaboration (e.g. Tang, J. C., 1991) that underpin the hypotheses. It evokes a mixture of independent and group work (Scott et al., 2004), which this study seeks to explore. The task is meaningful for the participants because they can apply prior experiences of room layouts. As a design task, it requires discussion and exploration of different approaches, and tradeoffs between requirements, by manipulating the task artefacts. Finally, because of the shared outcome, the tradeoffs in the brief, and the meaningful nature of the collaboration, this task does not lend itself to the divide-and-conquer approach observed in other tasks (e.g. Scott et al., 2005). Instead, participants must aim to remain aware of each other's actions if they are to produce a satisfactory outcome.

6.2.3 Technology and Interaction

Figure 6.2 shows the remote tabletop system used in the study. The horizontal interactive displays measured 70cm x 70cm and were set 6cm within the surface of a larger table. One display used a front-projected 6-projector tiled array, and the other a single front-mounted projector. Projectors were mounted at slightly oblique angles to prevent participants from being distracted by the shadows of their own hands, as suggested by Ashdown (2004). The intensities of the displays were adjusted to be comparable.

Both displays used Anoto styluses (described in Section 4.4) for input, and collaborators could interact concurrently in all conditions. Remote collaborators' arms were captured and displayed in the workspace using translucent arm shadows. Each collaborator's arm shadows were also displayed on their own tabletop in order to provide feedback of the remote representation.

At the outset of the task, the system presented the participants with a blue background containing an empty white floor plan, "piles" of diagrammatic furniture, a reminder of the task brief, and a key explaining the different furniture representations (Figure 6.3). The floor plan measured 49cm x 49cm and was empty except for lines marking different rooms.

All the task artefacts (including the plan) could be freely moved using the widely-adopted Rotate'N'Translate technique, which enables a task artefact to be simultaneously rotated and translated using a single stylus stroke (Section 2.2.2) (Kruger

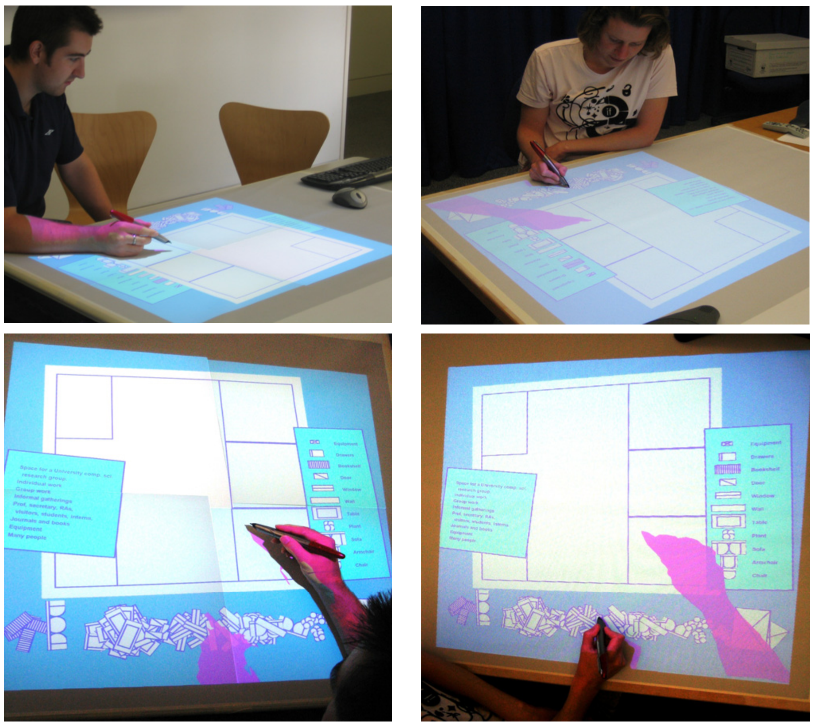


Figure 6.2: Tabletop interfaces in two different rooms (top left and top right) are linked to provide a large shared workspace (bottom left and bottom right).

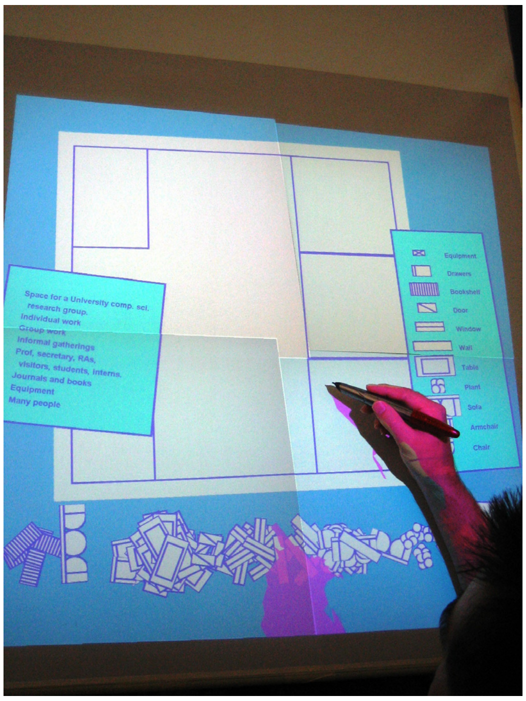


Figure 6.3: Initial view of the tabletop interface, showing floor plan (centre), reminder of brief (left), key (right) and piles of diagrammatic furniture (bottom).

et al., 2005). If the floor plan was moved, any furniture arranged on it would move with it. Based on feedback from a pilot study, the rotation mechanism was adjusted so that items of furniture on the floor plan snapped their relative orientations to multiples of $45°$. Task artefacts could be layered on top of other task artefacts by dragging, and dragging a partially occluded task artefact would bring it to the front.

The technical work described in the previous chapters provided the abstractions from which the furniture layout application was created, and mechanisms for display management, user input, remote arm representations and linking displays. The application performed at approximately 25fps with latency approximately 0.1s (measured using the procedure in Section 5.3). Arm shadows were captured and rendered at 12fps.

Polycom audio-conference speakerphones provided a full-duplex low-delay audio connection between the rooms using conventional telephone connections, with the volume adjusted appropriately.

6.2.4 Participants

18 participants aged between 20 and 39 were recruited from the department (Computer Science undergraduates and researchers) to form 9 pairs. 16 males and 2 females formed 7 same-sex pairs and 2 mixed-sex pairs. The partners in each pair reported having met each other previously. One participant reported limited previous experience using tabletop interfaces. 2 participants were left-handed, resulting in 7 right-handed-right-handed pairs and 2 right-handed-left-handed pairs. Each participant was paid £10.

6.2.5 Procedure

Participants signed an informed consent form and completed a demographic questionnaire. The consent form and all questionnaires referred to in this section can be found in Appendix C. The pair was then given a short tutorial about interacting with the system, after which each participant practiced individually until comfortable, which in all cases took around 2 minutes. Once the experiment structure had been explained, the pair completed each condition in turn.

In each condition, the participants were arranged appropriately and asked to stay in their seats and not to move the seats. For the remote conditions, both partners in each pair were shown both rooms. Participants practiced together until comfortable using a practice brief, to minimise learning effects during the recorded sessions. All pairs took about 10 minutes before their first condition, and about 2 minutes thereafter. The session workspace was then loaded. Participants were instructed to work together to arrange the task artefacts to fulfil the brief to the best of their abilities. They were advised that it might take between 15 minutes and half an hour, and that at the end they should make a short presentation and answer questions on their finished design.

After the presentation and questions, each participant completed post-condition questionnaire. Once all three conditions were complete, each participant individually completed a post-study questionnaire, and the pair took part in a semi-structured interview.

6.2.6 Problems Encountered

In two cases participants disrupted the calibration of the pen by leaning heavily on the table surface, causing it to shift. In these cases, the pair was asked to stop for about 2 minutes while the problem was fixed.

6.3 Results: Initial Observations

Pairs worked for an average of 21 minutes in each condition. Participants often used their prior experiences, such as that photocopiers should be situated away from working areas because of the noise. This led them to explore different layouts and adjust their design as they proceeded, to produce an outcome that was most appropriate in the context of their own prior experience.

Pairs in all three conditions seemed to have no difficulty interacting with the system or each other. Post-condition questionnaires asked about ease of task completion, ease of communication, and the extent to which the pair worked together, using 7-point Likert scales (Table 6.1). Analyses using Friedman rank tests for repeated-measures ordinal data yielded no significant differences between conditions. There is therefore no evidence that the conditions were perceived as different from each other in the extent to which they supported the collaborative task.

At the end of the study, participants were asked individually which of the remote conditions they preferred. 11 preferred overlaid, 6 preferred adjacent, and 1 had no preference. This indicates a trend towards a preference for the remote-overlaid condition, though a chi-squared test did not find that the preferences were significantly different from a 50:50 split.

Question	**CA**	**RO**	**RA**
"I was aware of what my partner was doing at all times."	1.8 (0.7)	2.4 (1.2)	3.1 (1.0)
"I was confident that my partner was aware of what I was doing at all times."	1.8 (0.5)	2.3 (1.0)	2.8 (1.0)
"It felt as if we were sitting at the same table."	1.1 (0.3)	2.7 (1.2)	3.5 (1.2)
"We worked together throughout the task."	2.0 (0.8)	2.1 (1.2)	2.1 (0.9)
"How easy or hard was the task to complete using this technology?" (1=very easy, 7=very hard)	2.5 (0.9)	2.2 (0.9)	2.4 (0.8)
"How did you find communicating this way?" (1=very easy, 7=very hard)	1.7 (0.8)	1.9 (0.6)	2.3 (0.8)

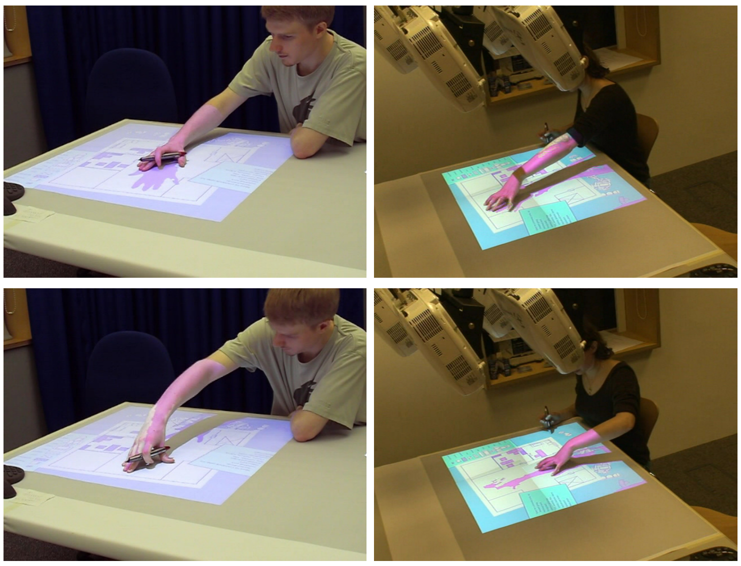
Table 6.1: Mean Likert scale responses (standard deviations in parentheses). For agreement questions, 1=strongly agree, 7=strongly disagree.

Excerpts from the recorded videos appear in this chapter as a series of still frames ordered from top to bottom accompanied by transcripts. These transcripts use an adapted version of the notation used by Kirk (2007) and Fraser (2000): L and R are collaborators left-most and right-most on the page; numbers in braces indicate video frames, so for example, {2} indicates the second frame from the top; underlined speech was accompanied by a gesture; (.) indicates a pause in speech; dashes indicate words cut short; and descriptions appear inside double brackets.

Participants used the arm representations to gesture to convey a variety of workspace gestures: deictic gestures, by pointing at a room or waving over an area (Figures 6.4 and 6.5); spatial gestures, using the hand to indicate shapes and trace paths (Figures 6.5 and 6.6); and action gestures to indicate, for example, turning (Figure 6.7). They seemed to have no difficulty interpreting these gestures.

6.4 Results: Coordination

This section reports the analysis of spatial partitioning as a coordination mechanism (Q3) and the effects of seating arrangement (Q4).

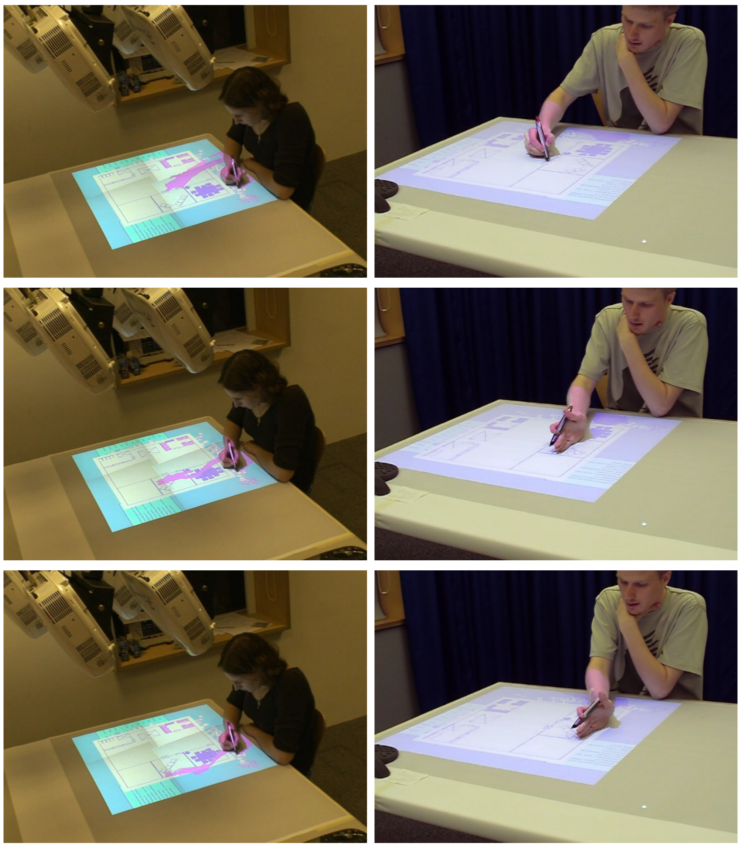


```
L: so the coffee machine's going to go here is that right?
R: yes so if we put like the informal bit up here {1} and the less the
more formal bit down here {2} with maybe some desks or something
```

Figure 6.4: Deictic gesture using the spread hand to indicate an area.

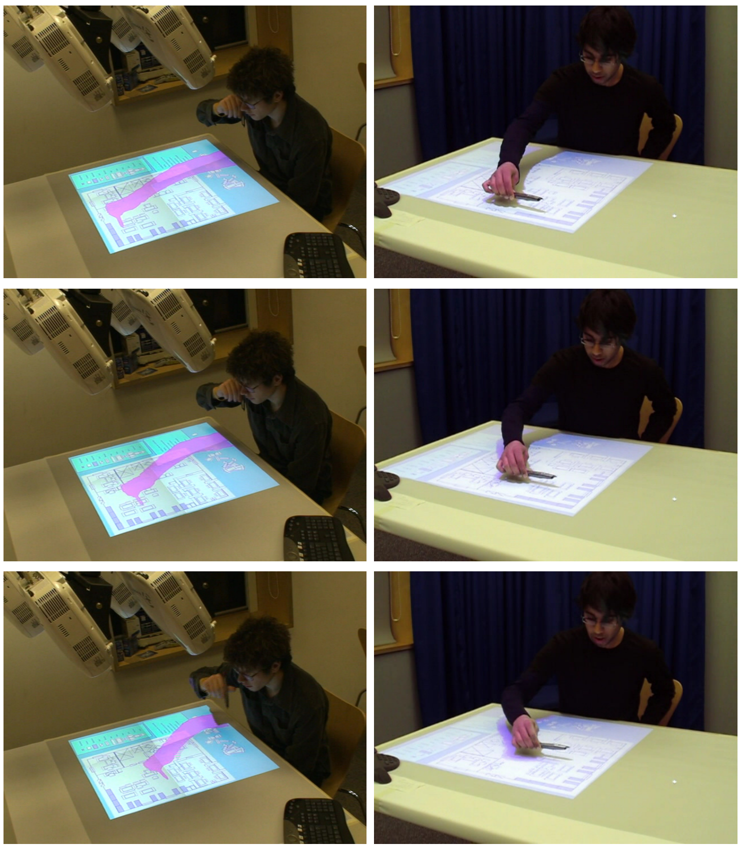
Studies of spatial partitioning have used a variety of analysis techniques. Tse et al. (2004) investigated a drawing task in which pairs used two mice to trace a shape on a shared screen. Spatial separation of action was inferred from the distribution of distances between the two mouse pointers during the task, and partitioning strategies were labelled by categories, such as "up/down", and "left/right". Scott et al. (2004); Scott (2005) investigated spatial partitioning in a tabletop furniture layout task on a conventional table. Workspace interactions were labelled by categories (e.g. "lifting", "rotating") and by table regions. This resulted in an *activity map* showing the distribution of activity by each participant in each table region. The observed trends informed a qualitative analysis of behaviour in each region. Ryall et al. (2004) investigated shared resources in an interactive tabletop poetry task and generated activity maps automatically from log files.

The spatial separation measure is not appropriate for this study, since the stylus location is unknown when lifted from the surface, and partitioning strategy labelling is too simplistic for an open-ended design task. Accordingly, this study uses activity maps.



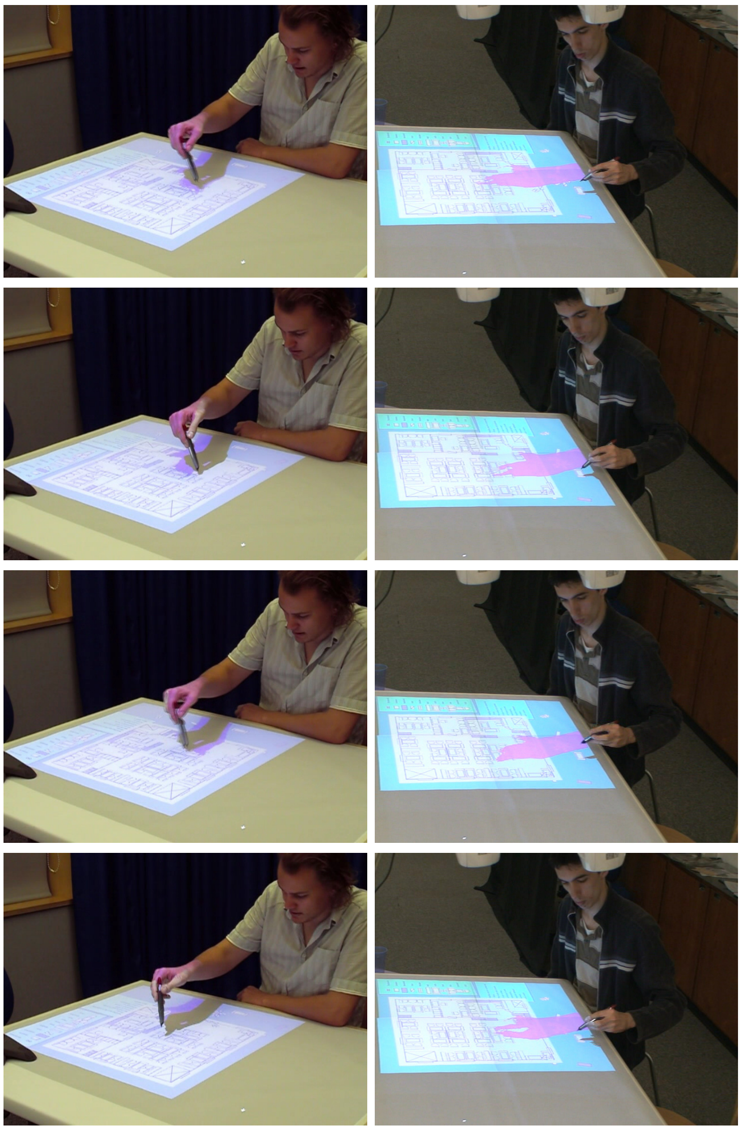
`R: if we have the tv like <u>here</u> {1} and then the chairs in banks like <u>here</u> {2} and <u>here</u> {3}`

Figure 6.5: Deictic gesture using the hand and stylus to point, and spatial gesture using the hand to indicate shape and orientation.



```
R: why don't we move these tables so they're like this {1, 2, 3}
L: oh yeah
```

Figure 6.6: Spatial gesture using the hand and stylus to indicate shape and orientation.



```
L: maybe we could flip this whole thing round 180 degrees ((makes
circling motion {1, 2, 3, 4}))
```

Figure 6.7: Action gesture using turning movement of the hand and stylus to indicate rotation

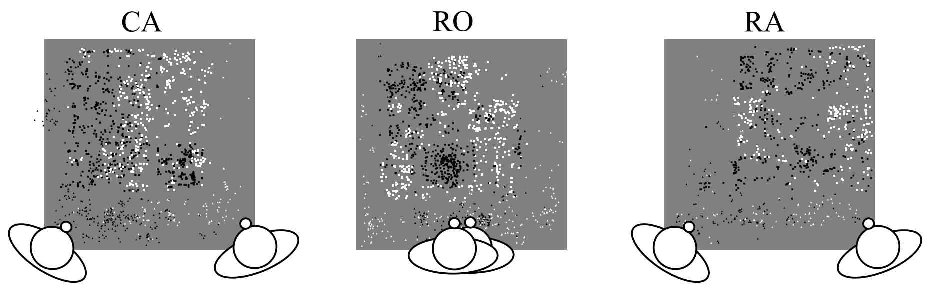


Figure 6.8: Activity map showing interactions of one pair in the workspace. Each point corresponds to a task artefact being picked up or dropped. Colour indicates the person interacting.

6.4.1 Visual Trends in Partitioning

Activity maps were generated from system log files and illustrate the locations of participants' interactions in the workspace as marks on a diagram. Figure 6.8 shows activity maps for a single pair for each condition. Further activity maps are shown in Appendix D. Although the floor plan was movable, its general position on the table is somewhat visible in the distribution of the markers. The marker distributions in the co-located-adjacent conditions show a trend for partitioning the floor plan and surrounding space according to proximity, so that each participant worked broadly in the half of the table nearest themselves. By contrast, the marker distributions in the remote cases suggests that participants broadly partitioned the workspace into regions in which they worked, but the partitioning forms a patchwork rather than a strict left-right arrangement.

6.4.2 Quantifying Left-Right Partitioning

Such a quasi-naturalistic open-ended task inevitably leads to variations in partitioning, depending on how activity unfolded at the time, and so it is difficult to make inferences from the visual data alone. Accordingly, a quantitative statistical analysis of the extent of left-right partitioning was performed on the underlying location data.

Plotting a histogram of x coordinates of each participants' interactions would allow comparisons between conditions using the mean and standard deviation.

However, the points are highly clustered, and it is unclear whether they can be modelled in this way. This study adopts an alternative approach, like Scott et al. (2004) of counting each participant's interactions in the left and right sides of the table. A derived numeric index measures the extent of partitioning between the two regions and collaborators, and was analysed statistically across pairs and conditions.

Scott et al. (2004) calculated the proportion of interactions in each region that were carried out by one particular participant in the pair (the left-hand region might, for example, have been 30% participant A and 70% participant B). However, this index is biased if one participant was more active overall. An alternative approach calculates the proportion of each participant's interactions in one particular region (participant A might, for example, have been 30% left and 70% right). However, such an index may also be biased if a single participant's value is considered in isolation from their partner's. If both participants interact solely on the right-hand side, for instance, then this index incorrectly indicates high partitioning for each participant. To avoid these problems, this study uses an index of partitioning *between* a pair of participants. For partners A and B, the index is computed by first calculating the proportions of A's interactions that lie within the left-hand side of the table, and then the same for B's, and then taking the absolute difference between these.

$$LRPI = \left| \frac{\#\text{ participant A in left region}}{\#\text{ participant A}} - \frac{\#\text{ participant B in left region}}{\#\text{ participant B}} \right|$$

"#" denotes "number of interactions of". The result remains the same if the right-hand side is used instead of the left. An index of 0.0 indicates no left-right partitioning (that both participants were equally likely to interact in a given side of the table), whereas an index of 1.0 corresponds to a strict left-right partitioning. This *left-right partitioning index* (LRPI) is robust even when one participant or one side is more active overall.

6.4.3 Quantified Left-Right Partitioning

I calculated the left-right partitioning index for each log file about the centre of the floor plan. Figure 6.9 shows the results for each condition. Error bars on the figure, and on all the figures in this chapter, are 95% confidence intervals and illustrate variation within pairs, disregarding baseline differences among pairs. Statistical analyses in a within-pairs study compare variation among levels of the independent variable against the remaining variation once baseline differences among

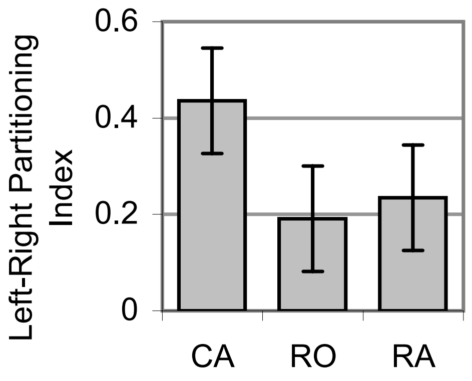


Figure 6.9: Left-right partitioning index for each condition. Error bars indicate 95% confidence intervals.

pairs have been accounted for. This yields the probability that variation among levels was caused by experiment noise. Such tests are sensitive in within-pairs studies because they explicitly account for and disregard the variation in baseline among pairs. Accordingly, it is legitimate to similarly disregard this variance when computing the confidence interval for error bars, following the procedure of Loftus and Masson (1994).

The results are consistent with the trends identified previously. In the co-located-adjacent condition, the mean LRPI of 0.44 corresponds to each participant carrying out 72% of their interactions on their own side of the table (and 28% on the other side of the table). In both the remote cases, regardless of the seating arrangement, the LRPI of approximately 0.2 corresponds to a 60%:40% split between the sides of the table. Curiously, this suggests that the degree of left-right partitioning in the remote-adjacent condition is comparable to that in the remote-overlaid condition, in which participants sat in the same position with respect to the workspace and hence were not able to partition by proximity at all.

A one-way repeated-measures analysis of variance showed a significant effect ($F(2, 16) = 7.02$, $p < 0.01$). Pairwise post-hoc tests were conducted using two-tailed repeated-measures t-tests and the Bonferroni-adjusted alpha level of 0.017 per test (.05/3). Results indicated that the LRPI was significantly higher in the co-located-adjacent condition than in the remote-adjacent condition ($t(8) = 3.64$, $p = 0.007$). The pairwise comparison between the co-located-adjacent and remote-overlaid conditions was approaching significance ($t(8) = 2.83$, $p =$

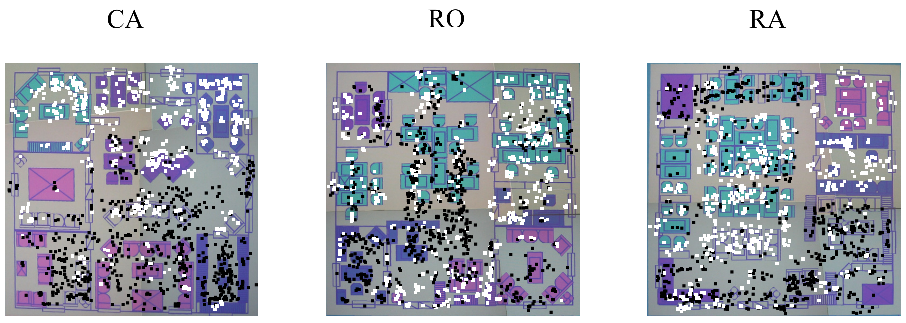


Figure 6.10: Interactions with furniture on the floor plan overlaid on the participants final outcome. The marker colour indicates the person interacting.

0.022). The comparison of the remote-adjacent and remote-overlaid conditions was non-significant.

6.4.4 Patchwork Partitioning

In order to further investigate the patchwork partitioning observed in the remote conditions, I transformed the marker locations into the frame of reference of the floor plan and superimposed them onto the participants' final furniture layout. Figure 6.10 shows this transformed view. Further transformed activity maps are shown in Appendix D.

In both remote conditions, collaborators partitioned the task broadly according to the room divisions that were visible as blue lines on the floor plan at the outset of the task. The floor plan also contained large open areas with no visible room divisions at the start of the task, which collaborators partitioned by creating islands of furniture and new walls.

6.4.5 Verbal Coordination

Participants in the remote conditions did not partition the space based on proximity according to social convention, as they had done when co-located, suggesting that they may have had to do extra work to coordinate their task activities.

In all conditions, much of the coordination seemed either implicit in the conversation and actions of the participants or dictated by social norms. However,

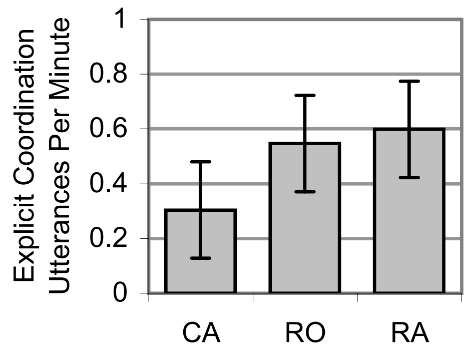


Figure 6.11: Rate of explicit coordination utterances for each condition. Error bars indicate 95% confidence intervals.

collaborators also used explicit coordination utterances to communicate to each other what they had done or should do, such as "I'll do the common room now". Coding the conversation allowed measurement of the rate of explicit coordination utterances in each session. A difference between conditions would provide further evidence of differences in coordination mechanisms.

Studies of remote collaboration have analysed conversation in various ways. Gutwin and Greenberg (1999) used verbal efficiency as an awareness measure in a controlled experiment using constrained tasks, but in an open-ended design task such a measure becomes meaningless by itself. Remote gesturing studies have used conversation analysis techniques to study conversational grounding (e.g. Gergle et al., 2004; Kirk et al., 2007). Again, this technique is not appropriate because this study is concerned not with the mechanics of gesture and grounding but rather the coordination of activities.

The dialogue was coded to identify coordination utterances that obviously matched one of the following:

- A participant informing the other participant what they intend to do (but not speculative planning), such as "I'll do the common room now", "I'm going to put a table in the office", "Shall I lay out the kitchen now".
- A participant directing the other participant what to do (but again, not speculative planning), such as "Why don't you lay out the kitchen", "You can start on the secretary's room", "You could do the common room now".

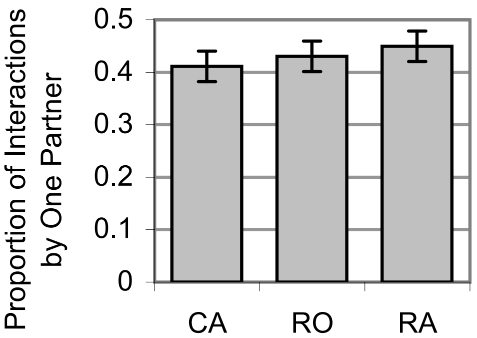


Figure 6.12: Distribution of work between partners for each condition. Error bars indicate 95% confidence intervals.

- A participant informing the other participant what they have done, such as "I've finished doing the windows", "I've moved the table round".

This scheme avoids attempting to identify requests for information (e.g. "Have you put the windows in?"). Although such utterances aid coordination, it was too difficult to reliably separate them from the general task discussion.

Figure 6.11 shows the rate of explicit coordination utterances per minute in each condition. Explicit coordination utterances were on average twice as frequent in remote conditions. A one-way repeated-measures analysis of variance showed that the effect was significant ($F(2, 16) = 6.14$, $p = 0.03$, using the Greenhouse-Geisser correction for poor sphericity).

Pairwise post-hoc tests were conducted using two-tailed repeated-measures t-tests with a Bonferroni-corrected alpha value of 0.017 per test. Results indicated that such utterances were significantly more frequent when co-located-adjacent than when remote-overlaid ($t(8) = 4.25$, $p < 0.01$). The comparison of the co-located-adjacent and remote-adjacent conditions was approaching significance ($t(8) = 2.40$, $p = 0.04$). The comparison between the remote-adjacent and remote-overlaid conditions was non-significant.

6.4.6 Work Distribution

I calculated the distribution of work between participants in order to investigate the effects of coordination differences on participants' roles and their ability to contribute. Figure 6.12 shows the distribution of interactions between participants in each condition. A value of 0.4 corresponds to a 40%:60% split of interactions between partners in a session.

A one-way repeated-measures analysis of variance was non-significant. The confidence intervals support direct observations that the condition seemed to have little practical influence on the roles of the participants or their ability to contribute.

6.4.7 Perceived Spatial Co-presence

The post-condition questionnaire asked a Likert-scale question to assess the perceived level of spatial co-presence, the feeling of "being in one place even when physically situated in another" (Witmer and Singer, 1998). Differences may provide insights into the differences in spatial partitioning between the conditions. Studies of virtual reality systems have used similar questions to assess their realism (e.g. Hauber et al., 2006). (Spatial co-presence is largely irrelevant in studies of conventional groupware systems and consequently has been ignored in these studies).

Table 6.1 (page 155) shows the results. A Friedman rank test for repeated-measures ordinal data showed a significant effect ($\chi^2(2) = 27.6, p < 0.01$). Pairwise tests were conducted using Wilcoxon signed rank tests, with a Bonferroni-corrected alpha of 0.017. The results showed a significant difference between all three conditions. Participants perceived a greater level of spatial co-presence when remote-overlaid than when remote-adjacent ($V = 80$, $p = 0.012$), a greater level when co-located-adjacent than when remote-overlaid ($V = 0$, $p < 0.001$) and a greater level when co-located-adjacent than when remote-adjacent ($V = 0$, $p < 0.001$).

6.5 Results: Workspace Awareness

This section reports the analysis of the postulated mechanisms for maintaining awareness (Q1a–Q1c) and the use of awareness to manage coupling (Q2). The use of intentional gesture (Q1a) in this study was reported in Section 6.3, and accordingly this section investigates the remaining mechanisms (Q1b and Q1c).

Studies of conventional groupware systems have inferred the level of awareness using measures such as perceived effort, verbal efficiency, completion time, work strategy, and user preference followed by semi-structured interviews (Gutwin et al., 1996; Gutwin and Greenberg, 1999). Such an approach is appropriate when using constrained tasks and when seeking to establish empirically the levels of awareness in order to compare different interfaces. However, these measures are less meaningful in a quasi-naturalistic open-ended design task. Furthermore, this section does not seek to establish some overall empirical level of awareness, but rather to investigate the processes by which awareness arises.

The analysis begins with identification of different observed coupling styles, ranging from independent working to working closely together. The video record was then coded to measure the proportion of time spent in each coupling style in each condition. This is indicative of the extent to which that condition impacted on their ability to work in that style. If, for example, one condition provided a much lower level of awareness in a particular coupling style, then collaborators would be expected to experience problems in that style and consequently to work more in other styles instead.

The analysis then examines qualitatively how the collaborators carried out various activities known to require workspace awareness in order to investigate whether the postulated awareness mechanisms contributed. Section 2.8.2 identified various such activities: demonstrating to others, and seeing visual feedback of instructions carried out; coordinating group activity when working closely together; anticipating the actions of others; offering assistance; and appropriate transitioning between coupling styles. This section also reports how weaknesses in the interaction design impacted upon these mechanisms, and finishes by reporting participant perceptions of awareness.

6.5.1 A Coding Scheme for Collaborative Coupling Style

My analysis of collaborative coupling follows the approach of Tang, A., et al. (2006b), who compared interaction techniques on a horizontal interactive display using a map-based route planning task. From a video record of the collaborations, they iteratively refined a coding scheme of 6 coupling styles, ranging from working closely together on the same problem, to one watching the other work, to working on different problems. By coding the entire video record using this scheme, they identified empirically how the different interaction techniques differed in their abilities to support the different coupling styles.

I iteratively refined a coding scheme for coupling styles observed in the present study. Randomly-selected segments of video data from each condition were repeatedly analysed using a similar approach to prior exploratory studies of co-located collaboration (Scott et al., 2004; Tang, A., et al., 2006b). The initial coding categories were informed by field notes and by the co-located tabletop coupling styles identified by Tang, A., et al. (2006b). This initial analysis yielded four distinguishable coupling styles that exhaustively and mutually-exclusively classify the coupling arrangement at any time:

Simultaneous work on the same problem, (SWSP). The collaborators are actively working together simultaneously to help each other solve the same problem, such as both arranging the windows or both arranging the same room. Conversation often accompanied this style.

View engaged. (As Tang, A., et al. (2006b)). The collaborators are working together but one is manipulating the display while the other watches closely. Participants arranged furniture to demonstrate ideas to each other, suggested corrections to each other, and also took turns to manipulate a particular arrangement of furniture. Conversation often accompanied this style.

Discuss. The collaborators are working together but are conversing and gesturing rather than manipulating the display. Discussion was often observed at the start and end of the tasks, for planning and reviewing the work.

Independent work. The collaborators are working independently, sometimes actively working and sometimes looking at the workspace. Collaborators often glanced at each other's work areas. The level of conversation in this style varied, independently of the seating condition, from periods of total silence, to occasional coordination utterances (such as "We're going to have a sofa by the pool table"), through to rapid chit-chat and full-fledged conversations.

6.5.2 Results of Coding

I analysed the entire video of each session to determine whether the conditions affected the tendency to collaborate in different coupling styles. The scheme was sufficient to describe the coupling style throughout every session. Figure 6.13 shows the results of this process.

Pairs spent considerable proportions of time in each of the coupling styles regardless of the condition. As might be expected with a quasi-naturalistic task, there was a large variation among pairs in the proportion of time spent in the coupling styles. Some pairs tended to stay closely-coupled throughout, whereas others worked mostly independently. Nevertheless, within each pair, the proportion of time spent in each style remained relatively constant, regardless of the condition. There were no trends to suggest that any condition promoted or inhibited a

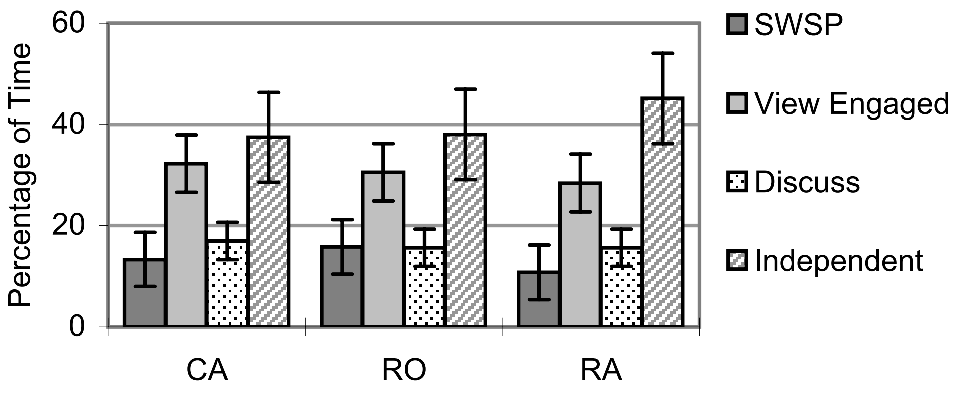


Figure 6.13: Proportion of time spent in each coupling style grouped by each condition. Error bars indicate 95% confidence intervals.

particular coupling style. One-way repeated-measures analyses of variance found no significant effects. No evidence was therefore found that switching between co-located and remote conditions affected the ability of the pairs to collaborate in different styles.

The frequency of transitions between coupling styles was not investigated, because the coupling styles are distinguishable only at a given temporal granularity. For example, participants conducting simultaneous work on the same problem would frequently pause for a second or two to watch each other or to discuss. Whether this constitutes a transition to a different coupling style depends on the level of granularity under consideration, and so transition frequency cannot be reported with confidence. This effect was not, however, problematic when measuring the proportion of time spent in each coupling style. Brief transitions to a different coupling style have a small impact on the results, even if the level of granularity were to vary slightly during coding.

6.5.3 Feedthrough and Consequential Communication

Coding the video provided qualitative insights into the relationship between coupling, and feedthrough and consequential communication. Most importantly, participants seemed able to transition among coupling styles rapidly and fluidly, regardless of the condition. Transitioning was opportunistic and seamless in all conditions, and instigated not by intentional gestures or conversation, but instead by workspace awareness from watching each other's arm movements (consequential communication) and manipulations of task artefacts (feedthrough).

Pairs working independently would frequently glance at each other's work. After glancing, one participant might then start to watch their partner (in the view-engaged style) before assisting them (in the simultaneous-work style) often by passing furniture to them or by arranging furniture in the same room. Their partner would recognise the transition and would make use of the passed furniture or begin to closely coordinate the arrangement. Transitions often occurred in silence, or while talking about something unrelated, relying instead on the consequential communication and feedthrough conveyed by the system. Figure 6.14 shows an example.

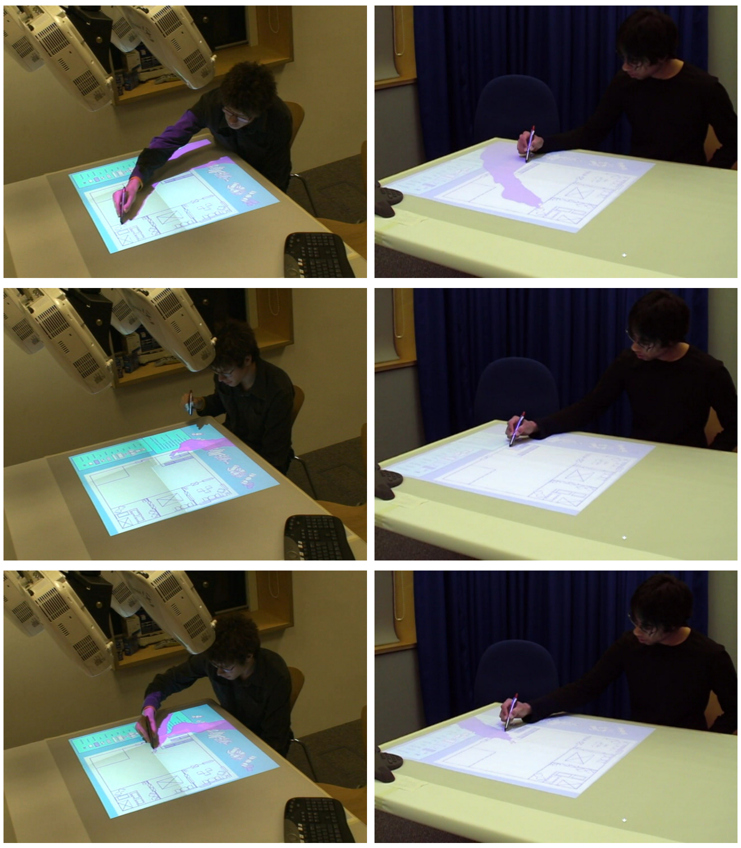
Collaborators would also transition from independent working to anticipate and assist in other ways, such as by watching and then providing verbal suggestions. This typically led into a session of turn-taking in which collaborators watched each other demonstrate different ideas. Again, the glancing and watching that preceded the transition and assistance relied on consequential communication rather than on verbal clarification. These transitions from independent working to assisting were observed in all conditions, even when collaborators were working in very disparate areas of the workspace. Figure 6.15 shows an example.

There were occasionally more direct uses of consequential communication, in which a participant would observe their partner's arm shadow reaching towards task artefacts or areas of the workspace and then make a suggestion based on this. Figure 6.16 shows an example.

6.5.4 Interaction Design

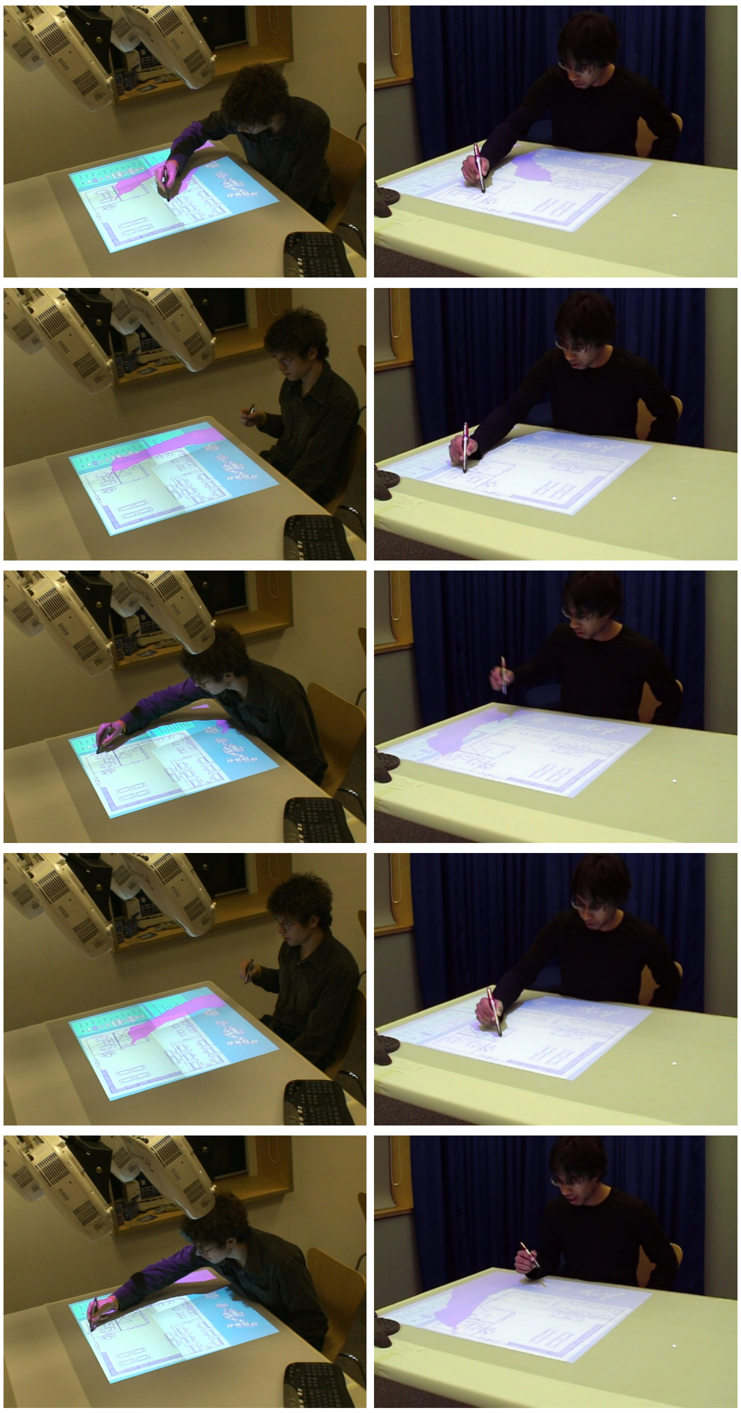
Although pairs transitioned fluidly and frequently throughout the vast majority of each session, there were also occasional incidents in which an action by one participant was completely unanticipated by their partner and resulted in confusion that was resolved verbally. These incidents were largely due to two weaknesses in the interaction design of isolated parts of the system. Analysis of these incidents provides design guidance for future interfaces, and also insights into how the otherwise-high level of workspace awareness was maintained.

Firstly, the large floor plan could be moved by touching any point on its surface with the stylus and then dragging. This action is incremental (small stylus movements produce small movements of task artefacts). However, a participant cannot



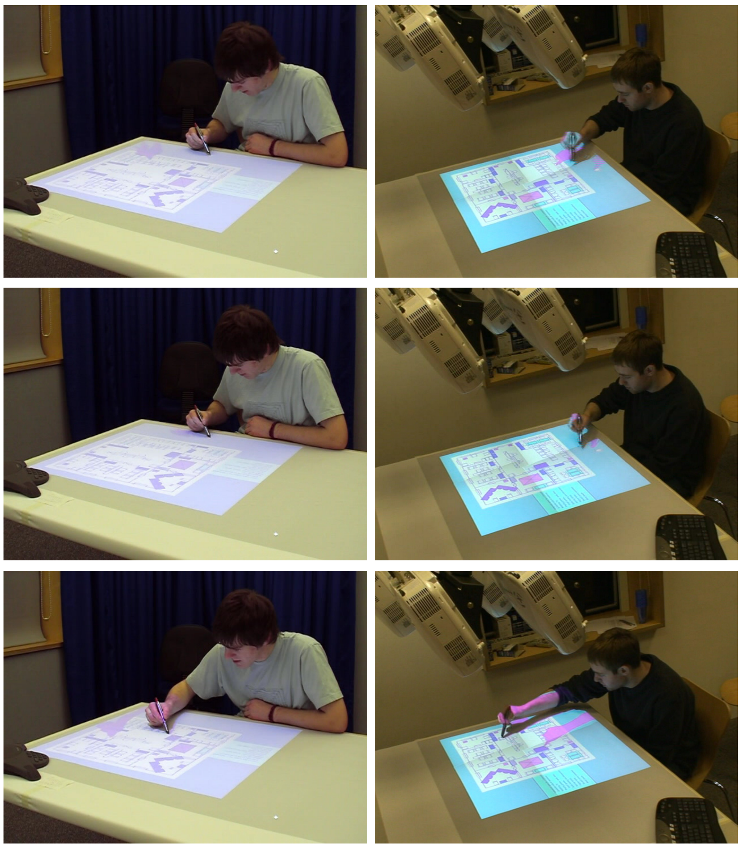
```
((Both are working independently)) {1}
((L watches R)) {2}
L: eh is good need a door
((L assists R)) {3}
```

Figure 6.14: Transitioning from independent work to view engaged to simultaneous work on the same problem.



((Both are working independently)) {1}
L: ((L watches R)) {2} ok we need windows
R: yeah ((R watches L put in a window)) {3}
R: put one in here as well ((L watches R put in a window)) {4}
R: they'll see the photocopier's in here so they know they can come and use it
L: ((L chuckles)) why?
((R watches L put in a window)) {5}
R: er it's nice having some light and air in the photocopier room

Figure 6.15: Transitioning from independent work to turn-taking.



```
((L is moving a piece of furniture near him)) {1}
((R’s stylus approaches the same piece)) {2}
L: no!
((R works elsewhere)) {3}
```

Figure 6.16: Anticipation based on consequential communication.

use the motion of their partner's arm to anticipate the initiation of dragging, because the floor plan is so large that its movement is not localised to the vicinity of the stylus. There were various incidents in which a collaborator deliberately moved the floor plan by dragging their stylus in one region, while their partner, working in another region of the plan, completely failed to anticipate the action. Figures 6.17 and 6.18 show examples.

Secondly, the system allowed participants to tap twice with the stylus to cause a moderately-sized colour menu to appear immediately, obscuring some of the workspace in the area immediately above the point of action. Although this action is reasonably localised to the vicinity of the interacting hand, it is symbolic rather than incremental (a small stylus movement causes a large menu to appear) and so it is not possible to see the action as it unfolds, either in the motion of the arm (consequential communication) or in the motion of the menu (feedthrough). This action by one collaborator was often unanticipated by their partner, which led to confusion. Figure 6.19 and 6.20 show examples.

In both cases, weaknesses in the interaction design prevented the actions from providing consequential communication and feedthrough, leading to confusion that was resolved verbally. These infrequent incidents highlight the need for appropriate interaction design and also the extent to which the system conveyed consequential communication and feedthrough during the vast majority of the sessions.

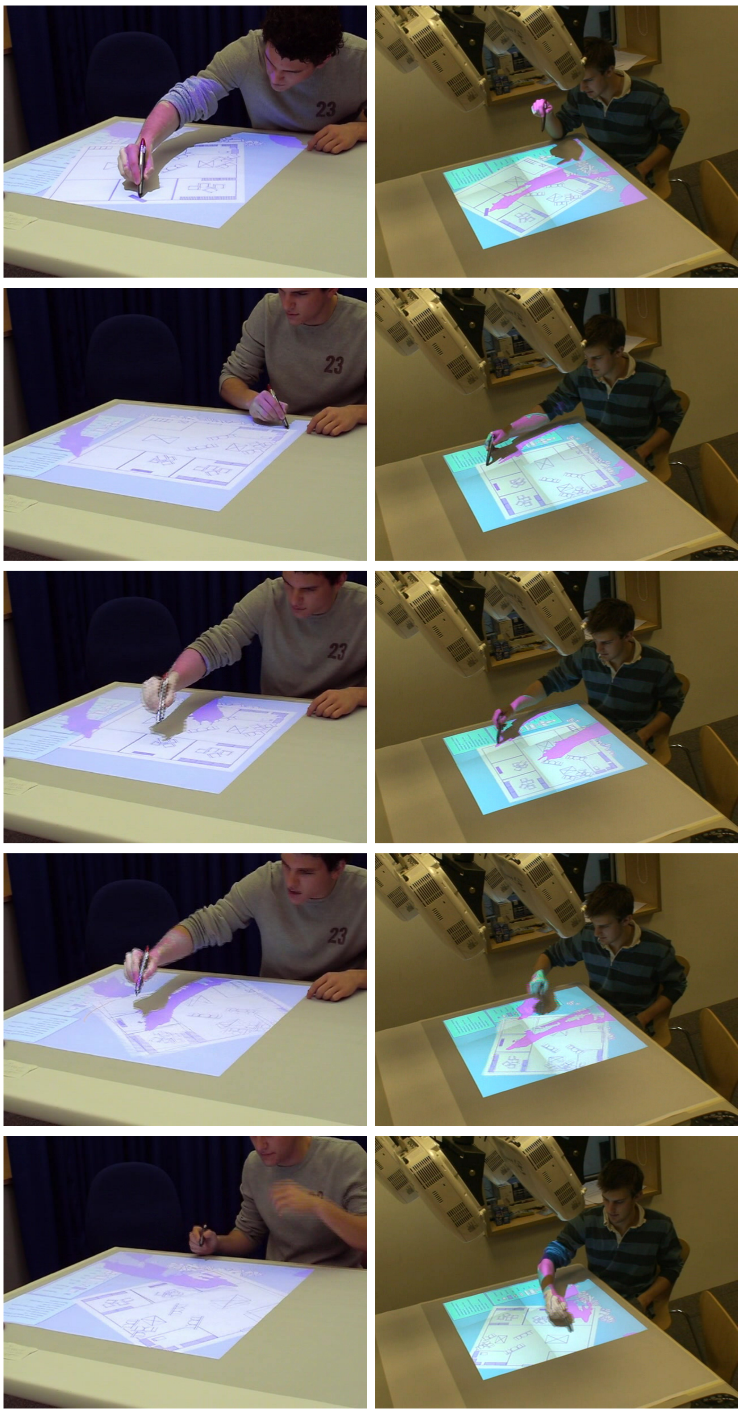
6.5.5 Perceived Awareness

Following the approach of Apperley et al. (2003), participants answered two Likert-scale questions about perceptions of awareness in each condition (Table 6.1, page 155). Friedman rank tests for repeated-measures ordinal data showed significant effects in responses to both questions ($\chi^2(2) = 14.00$, $p < 0.01$; and $\chi^2(2) = 12.04$, $p < 0.01$, respectively).

Pairwise comparisons were conducted using Wilcoxon signed rank tests with a Bonferroni-corrected alpha of 0.017. The responses to both questions showed significant effects between the co-located-adjacent and remote-adjacent conditions ($V = 0$, $p = 0.001$; and $V = 10$, $p = 0.004$, respectively). The remaining comparisons were non-significant.

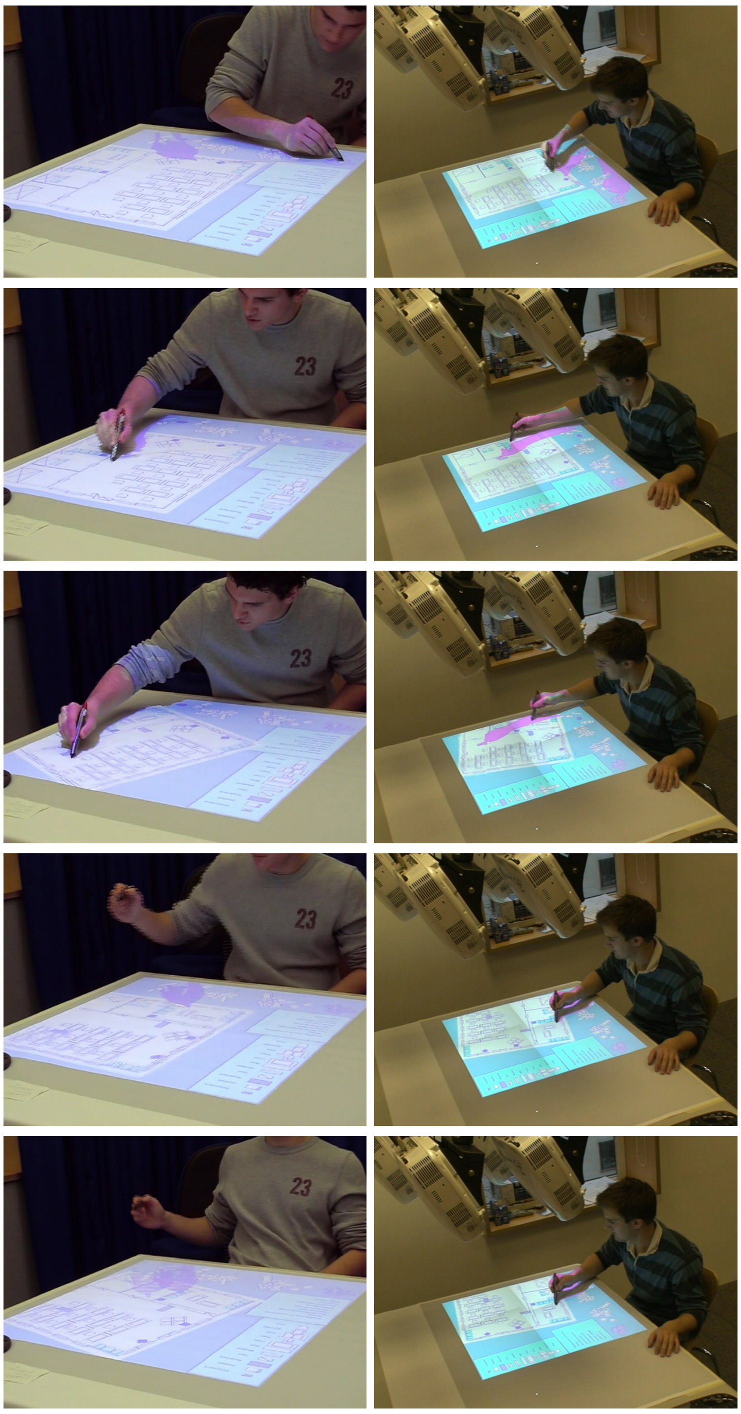
6.6 Discussion

This exploratory study set out to investigate remote tabletop collaboration, and in particular the proposed awareness mechanisms (Q1a–Q1c), collaborative cou-



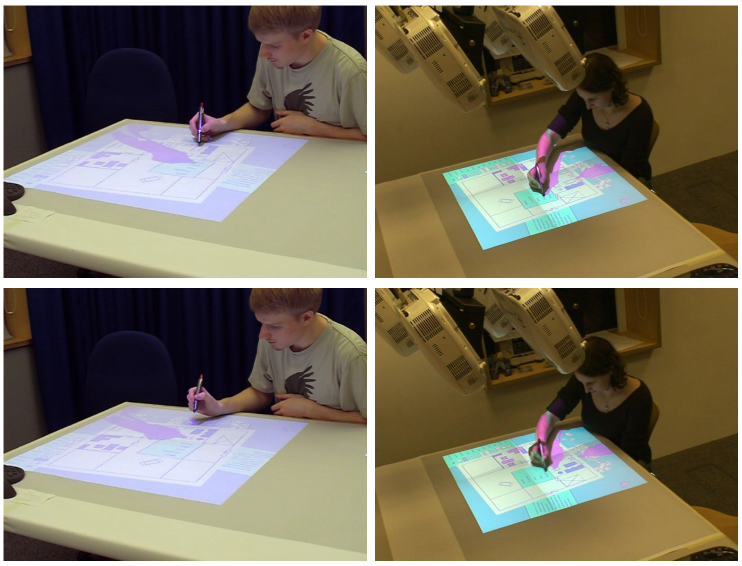
```
((L is working at the top of the plan)) {1}
R: do you want to spin the plan so you can get at it more easily
((L spins the plan)) {2}
((L reaches back towards the top of the plan start working)) {3}
((R spins the plan.  This is not anticipated by L, who backs off and
looks puzzled)) {4, 5}
L: oh i see yeah
```

Figure 6.17: Deliberate movement of the floor plan by one participant was unanticipated by their collaborator.



```
R: we haven't actually managed to put any shelving up
L: er (.)  which ones are the shel- ah here we go
((L begins to drag a piece of furniture)) {1, 2}
R: ah wait i'm going to spin the plan
((R spins the plan.  This wasn't anticipated by L, who stops dragging the
piece and sits back)) {3, 4, 5}
R: sorry
```

Figure 6.18: Deliberate movement of the floor plan by one participant was unanticipated by their collaborator.



```
((R has opened a menu and is about to press on it with the stylus)) {1}
((L opens a menu as R presses, causing R to press on the menu opened by
L)) {2}
L: oh?
```

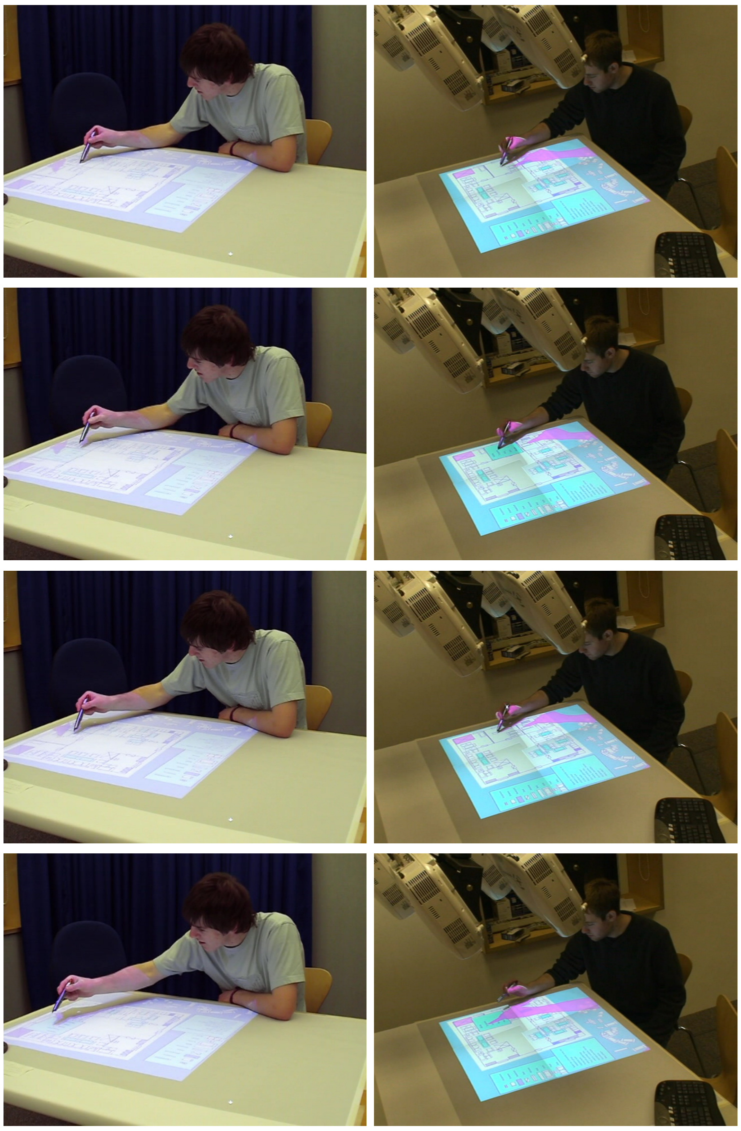
Figure 6.19: Opening of a menu by one participant was unanticipated by their partner.

pling (Q2), territorial coordination (Q3) and the effects of the overlaid seating arrangement (Q4). This discussion summarises and explains the results. The implications, limitations, and a programme of future research are described in the following chapter.

6.6.1 Preference of Remote Seating Arrangement

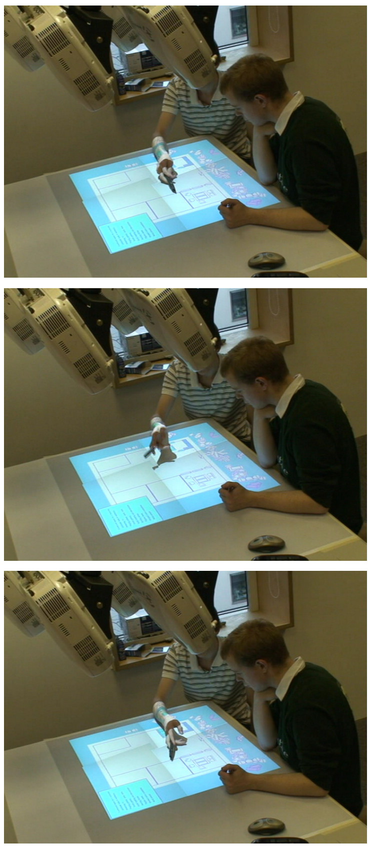
Like Kirk (2007), this study found a non-significant preference for the overlaid arrangement over a non-overlaid arrangement. Further analysis revealed two trends. Firstly, 10 of the 11 participants who preferred the remote-overlaid condition stated in questionnaire responses that they had difficulty reaching areas of the surface in the remote-adjacent condition. Curiously, none mentioned this in the co-located-adjacent condition, in which they were sat in the same positions.

This difference may be due to a limitation of the remote gesture system. Co-located-adjacent participants could point to parts of the surface far away from themselves without having to lean. The hand would be positioned some distance



```
((Both are working in the same part of the plan))
((L opens a menu that obscures R’s work)) {1}
((L presses on the menu and it closes)) {2}
((L again opens a menu that obscures R’s work)) {3}
((L presses on the menu and it closes)) {4}
R: ((tuts)) don’t do that while i’m trying to move stuff otherwise i
can’t move anything
L: ah excuses!
```

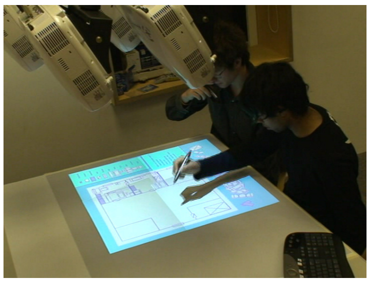
Figure 6.20: Opening of a menu by one participant was unanticipated by their partner.



```
L: I think you probably want a separate room for television and {1} pool
{2} and board games {3}
```

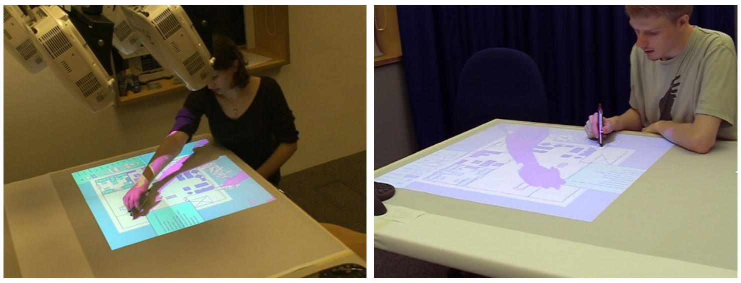
Figure 6.21: Three-dimensional deictic gesture in the co-located-adjacent condition.

vertically above the workspace, with the referent along the three-dimensional trajectory of the finger (Figures 6.21 and 6.22). However, the remote conditions did not convey these three-dimensional pointing gestures, since the gestures are displayed remotely on the surface with no depth cues. Participants in the remote conditions seemed aware of this and kept their hands close to the table surface when gesturing. In the remote-adjacent condition, this sometimes necessitated leaning, which may have caused frustration (Figures 6.23, 6.24 and 6.25). I speculate that this reaching may also cause fatigue in a longer task, or when using an input technology that prohibits the resting of the elbow on the display when reaching out (such as a multi-touch surface). They were not observed trying and failing to use three-dimensional pointing gestures in the remote conditions.



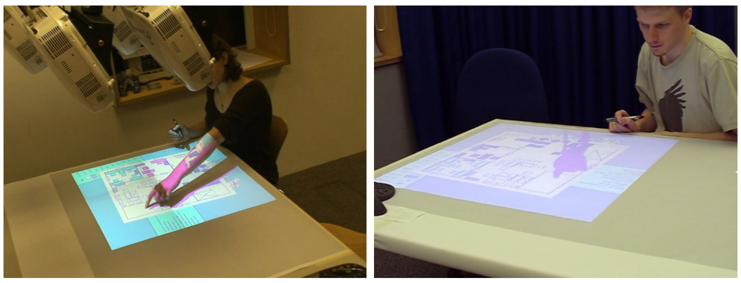
`R: I was thinking we should` <u>`make this room a meeting room`</u> `{1}`

Figure 6.22: Three-dimensional deictic gesture in the co-located-adjacent condition.



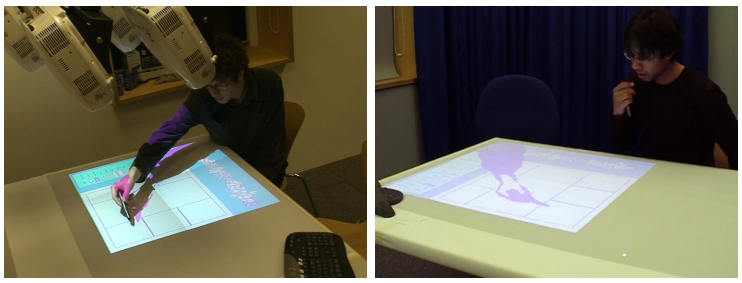
`L: or in` <u>`here`</u> `{1}`

Figure 6.23: Leaning to perform deictic gesture in the remote-adjacent condition.



`L: so what's going in` <u>`here`</u> `{1}`

Figure 6.24: Leaning to perform deictic gesture in the remote-adjacent condition.



`L: this might be the serving drinks one {1}`

Figure 6.25: Leaning to perform deictic gesture in the remote-adjacent condition.

By contrast, 5 of the 6 participants who preferred the remote-adjacent condition stated in questionnaire responses that the (partial) opacity of their collaborator's arm in the remote-overlaid condition caused problems. This, and the non-significant overall preference for the remote-overlaid arrangement, leads me to speculate that in a natural setting where they have a free choice, and well-tuned opacity, remote tabletop collaborators would choose the overlaid position. This in turn would have implications for territoriality and also for other tabletop coordination mechanisms such as the orientation of task artefacts, as discussed previously.

6.6.2 Coordination

I investigated territorial partitioning of space (Q3) and the effects of seating arrangement (Q4) using direct quantitative comparisons between the experimental conditions.

The study condition significantly affected the collaborators' partitioning of the workspace. Participants in the co-located-adjacent condition partitioned the floor plan and surrounding space broadly according to who was nearest. This proximity partitioning is consistent with the territoriality findings of Scott et al. (2004). Their study involved two separate sessions involving two co-located participants and quotes the number of interactions by region and by participant. The LRPIs from their data can be calculated as 0.51 and 0.42, which agree with this study.

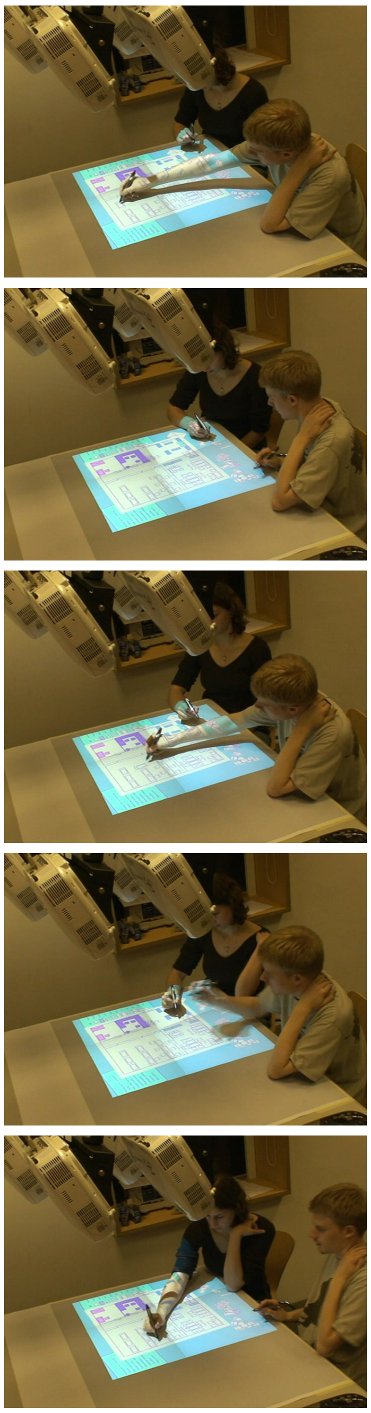
Participants in the remote-overlaid did not have the opportunity to partition and delegate on the basis of proximity. Nevertheless, each pair consistently partitioned the floor plan into areas in which one or the other participant worked, using either the boundaries visible at the start of the task, or by planning islands of task artefacts in which to work.

It is perhaps surprising that participants in the remote-adjacent condition behaved much as those in the remote-overlaid condition. The adjacent seating arrangement did not lead to the proximity-based partitioning observed in co-located-adjacent collaboration. This indicates that proximity partitioning arises from more than seating arrangement alone. Observations suggest that the difference is linked to collaborators' abilities to reach parts of the workspace without stopping their partner from working. Participants in the co-located-adjacent condition seemed wary of reaching across their partners to an area of the floor plan, or taking furniture from immediately in front of their partner. They would typically wait for an opportune moment, and sometimes ask permission (Figures 6.26 and 6.27). In the remote conditions, by contrast, participants would frequently work across each other, work within the shadows themselves, and take furniture from immediately in front of each other, without hesitation or utterance (Figure 6.28 and 6.29). These behaviours go some way to explaining the differences in partitioning.

However, other results suggest that the patchwork partitioning in the remote-adjacent condition cannot entirely be attributed to the ability of partners to work across each other. Of all the conditions, the remote-adjacent condition caused participants to feel least like they were sat at the same table. Participants also reported frustrations at the extra effort required to reach parts of the table when remote-adjacent, and yet the activity maps show that they nevertheless worked in those areas of the table. This suggests that participants were entirely non-territorial; that they viewed the entire space as in some way their own, working in whichever free space they wanted to, and were frustrated when it was harder to reach.

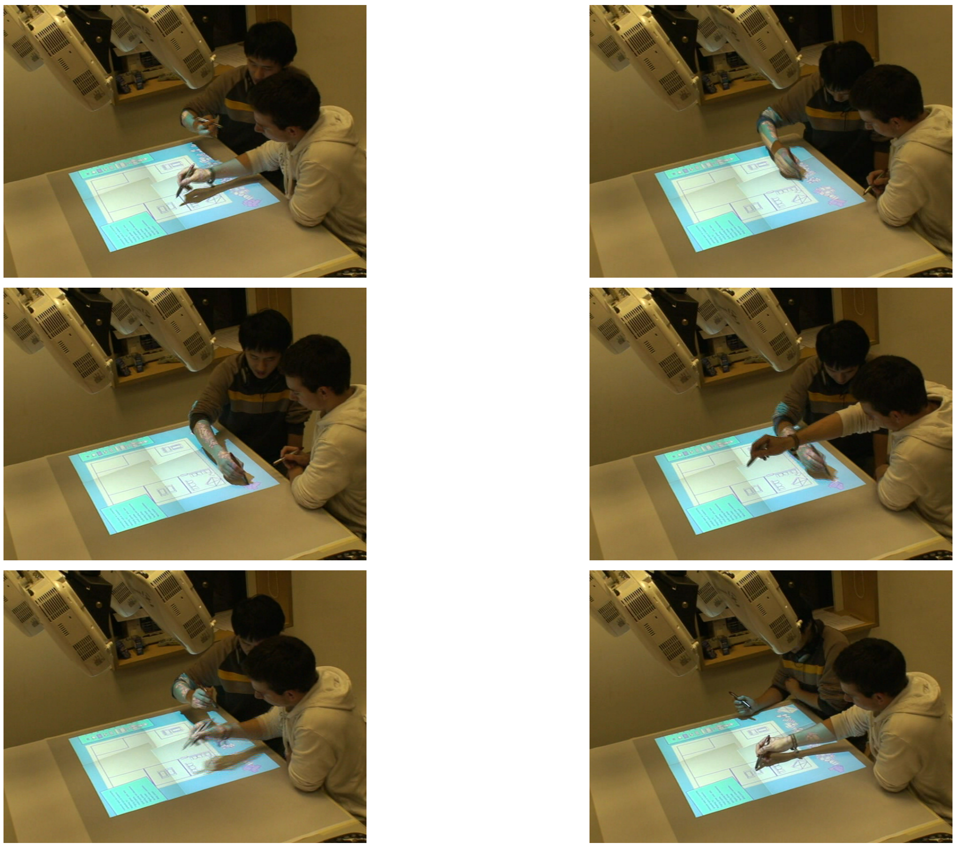
The patchwork partitioning observed in the remote conditions coincided with a greater rate of explicit verbal coordination. This suggests that patchwork partitioning, while less constrained than proximity partitioning, requires greater effort to coordinate activities. There was no evidence that the conditions affected the roles of the participants or their ability to contribute.

A more acute effect of the patchwork partitioning is the impact on the wider notion of territoriality. Section 2.3.2 described how spatial partitioning by proximity is a coordination mechanism beyond mere partitioning of the task, and serves as a means to reserve space and task artefacts. The remote-overlaid condition does not permit proximity partitioning. That collaborators in the remote-adjacent condition happily took items of furniture from in front of each other without hesitation or utterance suggests that they also are not able to use these mechanisms.



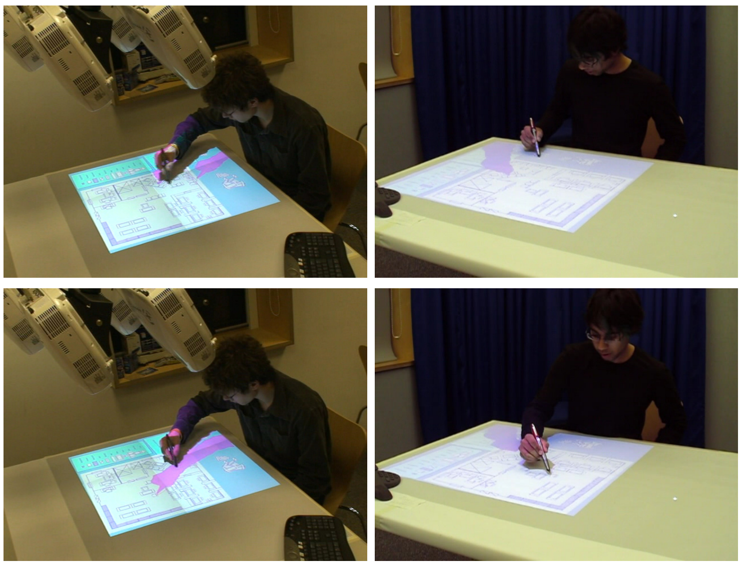
```
((R’s arm is preventing L from reaching the far side of the plan)) {1}
((R withdraws the arm but L is not ready)) {2}
((R works and again the arm prevents L from reaching)) {3}
((As R withdraws the arm, L reaches out)) {4}
((L reaches the far side of the plan)) {5}
```

Figure 6.26: Collaborators have difficulty working across each other when co-located-adjacent.



```
((R and L are working independently)) {1}
L: yeah I'll start doing pigeon holes
R: okay
((L leans across and takes a bookshelf while R is leaning back)) {2}
L: um bookshelves are pigeon holes
((R leans forward to work)) {3}
L: can I just steal like loads?
((R leans back and L leans across to take more bookshelves)) {4}
((R tries to work)) {5}
((L finishes taking shelves and R leans forward to work again)) {6}
L: cheers
```

Figure 6.27: Collaborators have difficulty working across each other when co-located-adjacent.



```
R: maybe a couple of these small chairs
((R starts to drag a chair)) {1}
((R drags it through L's hand.  L does not reply)) {2}
```

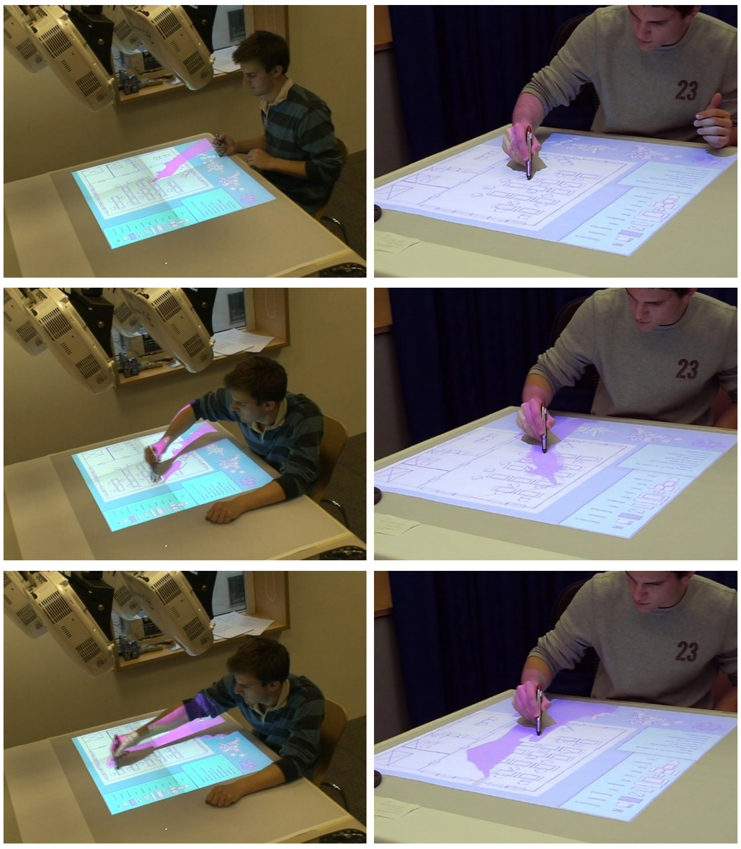
Figure 6.28: Collaborators work across each other when remote.

6.6.3 Workspace Awareness

I investigated awareness mechanisms (Q1a–Q1c) and collaborative coupling (Q2) using quantitative comparisons of coded video, supported by observations.

Four distinct coupling styles were identified, ranging from working closely together on the same problem to working independently. No significant differences among the conditions were found in the proportion of time spent in each coupling style. These results contrast with a study by Tang, A., et al. (2006b), which coded for similar coupling styles and found that varying the tabletop interface led to large differences in proportions of time spent in different styles.

One cannot experimentally prove a similarity in this way. Statistical techniques can show that distributions of measurements in different conditions are equivalent within some defined bound (though not the hypothesis-testing techniques employed in this study), and yet it may be the case that a difference would be more pronounced given a more demanding task. Nevertheless, for this representative design task, the confidence intervals show that changing among the conditions had little practical impact on collaborators' behaviour at this aggregate level. This in turn suggests that there was no practical impact on collaborators' abilities to



```
((L and R are working independently)) {1}
((L reaches up to the top of the plan, covering R's work with his own
arm)) {2}
((L's arm remains there for a few seconds.  R does not comment))
```

Figure 6.29: Collaborators work across each other when remote.

work in any of these styles, which is indicative of similar levels of workspace awareness.

This similarity is supported by participants' positive perceptions in all conditions of ease of task completion, communication, and extent to which they worked together. When asked directly about perceptions of workspace awareness, participants' responses were also positive in all conditions, but significant differences were found between the remote-adjacent and co-located-adjacent conditions. Post-study interviews did not yield reasons for this effect, and the difference was not investigated further. Section 2.6 described how co-located collaboration supports many visual cues that are not conveyed by the remote tabletop system, and so such a difference is perhaps not surprising.

I took a different approach to investigate the hypothesised awareness mechanisms (Q1a–Q1c). Suppose these research questions were to be studied by testing hypotheses between conditions in the usual quantitative way. This would involve comparing a remote tabletop system that followed the design guidelines to some other shared workspace interface that did not, perhaps a conventional GUI-based system. Such an experiment would allow broad qualitative comparisons of behaviour, but it seems unlikely that effects could be measured and compared quantitatively in such different systems. Instead, this study adopted a different approach, using qualitative observations to support or refute the hypothesised awareness mechanisms.

In support of Q1a, participants successfully communicated using pointing, spatial and kinetic gestures via the remote arm representations. As described in Section 3.1, spatial and kinetic gestures are not as easily conveyed by more conventional telepointers or traces. This supports G1, and corroborates results by Tang, A., et al. (2006b) and Kirk et al. (2005).

Examination of the video record focused on the mechanisms by which participants accomplished several activities that rely on workspace awareness: frequent, fluid transitions between coupling styles, fluid turn-taking, demonstrating to each other, anticipation and assistance, and close coordination when working together simultaneously on the same problem (Gutwin and Greenberg, 2002). Video observations showed how these activities relied on consequential communication and feedthrough, which in turn arose from the combination of arm representations, direct input styluses, and the direct manipulation, localised to the vicinity of the interacting hand, of the furniture task artefacts. Isolated weaknesses in the interaction design occasionally caused this process to break down, and further highlighted the importance of local, direct manipulation. The movement of the large floor plan was not local to the hand and could be instigated from anywhere on it, and so collaborators were not able to anticipate the instigation of the action.

The double-tapping action to open the menu was symbolic rather than incremental and again was often unanticipated. These breakdowns contrast with the fluid transitions between coupling styles that occurred during the vast majority of each session, which were mediated by the local, direct manipulation of the furniture task artefacts. These results provide support for the proposed awareness mechanisms Q1b and Q1c, and guidelines G2–G4.

The awareness contribution of avoiding workspace navigation (Q1d) was not investigated directly, and so the results offer little support. Pairs in the remote conditions spent similar proportions of time working independently to when they were co-located, and were able to transition fluidly from independent working to other styles, for instance to offer assistance, even when working in disparate areas of the workspace. As described in Section 3.1, it is more difficult to maintain awareness in such circumstances when using scrolling-based workspace navigation.

Accordingly, the findings support the awareness mechanisms Q1a–Q1c and, therefore, some of the design guidelines for remote tabletop interfaces developed in Chapter 3:

G1. Display high-fidelity arm representations in the workspace.

G2. Use direct input devices such as styluses or direct touch.

G3. Interaction should follow direct manipulation principles to result in immediate incremental, continuous visual changes, such as dragging of task artefacts.

G4. Interaction should be localised to the vicinity of the hand, such as dragging of small task artefacts.

In summary, the findings suggest that remote tabletop interfaces with movable task artefacts are beneficial not because they support the territoriality observed in co-located collaboration, and not because they provide a metaphor for real-world task artefacts, but because they provide a high level of awareness through consequential communication and feedthrough.

6.7 Chapter Summary

In this chapter I set out to explore empirically awareness and coordination in remote tabletop collaboration, in order to investigate some of the design guidelines and questions developed in Chapter 3. The study has found similarities with co-located tabletop collaboration in the ability to work in a variety of coupling styles and to transition fluidly between them. This relies on a high level of workspace awareness, and the study showed that this arose from the hypothesised awareness mechanisms. However, there are fundamental differences between co-located and remote tabletop collaboration in the way collaborators partition the space and prefer to sit. These findings highlight the potential of remote tabletops as collaboration interfaces, and support some of the design guidelines developed in Chapter 3. Their implications are discussed in the following chapter.

Chapter 7

Conclusion

This chapter concludes the dissertation. I begin by summarising each of the contributions, and then extend them beyond the immediate setting of the dissertation to discuss both their limitations and the new opportunities presented.

7.1 Summary of Contributions

This work was motivated by the potential benefits of applying a tabletop interface approach to the problem of remote collaboration. It builds on previous work in the fields of HCI and CSCW to make three main contributions to knowledge about remote tabletop interfaces: a theoretical basis for their design; a method for their construction; and findings about their use in practice. This section summarises each contribution in turn.

7.1.1 Design of Remote Tabletop Interfaces

The review and discussion of the literature established shortcomings in the theoretical basis and design of remote tabletop interfaces (Chapter 2). The review began by considering tabletop interfaces for co-located collaboration. They tend to present virtual task artefacts that can be moved, rotated, and grouped. These interaction techniques were motivated by mimicry of physical task artefacts and the need to support collaborators sitting around the table. Such interfaces can support a range of beneficial work practices that have been observed in tabletop collaboration with physical media, such as the use of spatial partitioning as a coordination mechanism, and support for fluid transitioning between working independently and working closely together. By contrast, the review of prior remote tabletop interfaces established that the underlying theoretical basis was unclear and that there was little empirical evidence to support the approach.

I then built on the work of Tang, A., et al. (2006b) to apply the workspace awareness framework of Gutwin and Greenberg (2002) as a theoretical basis for the design of remote tabletop interfaces (Chapter 3). Provision of a high level of workspace awareness is a prerequisite for many beneficial collaborative work practices. Awareness of the actions of one's collaborators in the workspace is rather assumed in co-located tabletop collaboration, but must be explicitly considered in the design of remote collaboration technologies that convey only a limited range of awareness cues. This process resulted in several mechanisms by which large horizontal interfaces and, in particular, tabletop interaction techniques might offer a high level of workspace awareness in remote collaboration:

Q1a. High-fidelity representations of arms, displayed in the workspace, enable rich gesturing for intentional communication.

Q1b. Direct manipulation techniques enable task artefact feedthrough.

Q1c. High-fidelity representations of arms, displayed in the workspace, together with direct input devices, enable consequential communication when used with local, direct manipulation techniques.

Q1d. Avoiding workspace navigation, by using large displays and appropriate interaction, enables collaborators to work independently in different areas of the workspace while maintaining workspace awareness.

This analysis also resulted in the following proposed design guidelines for remote tabletop interfaces:

G1. Display high-fidelity arm representations in the workspace.

G2. Use direct input devices such as styluses or direct touch.

G3. Interaction should follow direct manipulation principles to result in immediate incremental, continuous visual changes, such as dragging of task artefacts.

G4. Interaction should be localised to the vicinity of the hand, such as dragging of small task artefacts.

G5. Avoid workspace navigation by using large displays and appropriate interaction design, such as casual piling and shrinking.

An exploratory study (Chapter 6) observed that mechanisms Q1a–Q1c contributed to workspace awareness, and so supports design guidelines G1–G4. The study did not investigate Q1d and G5.

The design guidelines and theoretical motivations provide a basis for the further investigation of interaction techniques, applications and work practices in remote tabletop interfaces, and also inform their technical construction.

7.1.2 Construction of Remote Tabletop Interfaces

The literature review (Chapter 2) and subsequent analysis (Chapter 3) established further that remote tabletop interfaces present technical challenges because of the need to combine both tabletop interaction techniques and remote collaboration. Whereas co-located tabletop collaboration interfaces can be implemented using reusable systems such as DiamondSpin (Shen et al., 2004), no such system exists for remote tabletop interfaces. Resolution of this issue may enable more rapid exploration of applications and interaction techniques for remote tabletop interfaces, and may go some way to addressing the rather limited range of remote tabletop applications that has so far been demonstrated. The review identified that, although the current applications, such as jigsaw puzzles, enable investigation of collaborative work practices, there has been little investigation of the kinds of applications for which people currently use their desktop computers and meeting-room tables. Collaborative data analysis, document review and web browsing, for instance, are activities that remote tabletop interfaces could potentially benefit. I identified the low spatial resolution of current tabletop displays as a considerable technical barrier to the construction of such systems, and established that reuse of existing user interface components is also important to enable rapid prototyping (Chapter 3).

I then reviewed and implemented methods to create a novel system that addresses these problems to enable researchers to more rapidly implement a wider range of remote tabletop applications (Chapter 4). This contributes a method that enables: a large shared workspace for remote collaboration using tabletop interaction techniques; a novel method for segmenting arms in a camera image using a front-projected display; high-resolution displays using a tiled array of projectors and techniques from multi-projector display walls; and a useful programming interface for creating applications that enables rapid prototyping through the reuse of existing user-interface components. An evaluation of these subsystems has characterised their capabilities and limitations, and identified areas for future technical improvements (Chapter 5). A range of novel applications created using the system demonstrates the utility of the method (Chapter 5).

7.1.3 Work Practices in Remote Tabletop Collaboration

The discussion of the literature identified work practices that have been observed in co-located tabletop collaboration with both digital media and physical media (Chapter 2). I hypothesised that remote tabletop interfaces could potentially afford these practices: frequent, fluid transitions between a range of coupling styles; spatial partitioning as a coordination mechanism; and orientation of task artefacts as a coordination mechanism. I further identified that remote tabletop collaborators can sit in an overlaid arrangement, and that this potentially affects the coordination mechanisms of spatial partitioning and orientation (Chapter 3). I conducted an exploratory study to investigate the awareness mechanisms, the work practices of coupling transitions and spatial partitioning, and the effects of seating arrangement (Chapter 6). The study compared co-located tabletop collaboration and remote tabletop collaboration using an overlaid and a non-overlaid seating arrangement.

The study results inform the design of remote tabletop interfaces and highlight areas for their future study. They suggest that a considerable proportion of remote collaborators prefer the overlaid seating arrangement. Co-located collaborators partition the space broadly according to who is nearest, and hesitate or ask before reaching across each other. Previous studies have found that this behaviour serves as a coordination mechanism (see Section 2.3.2). Remote collaborators, however, do not partition the space in this way, regardless of the seating arrangement. Instead, they partition based on visual markings in a patch-work arrangement, seemingly with little regard for where they are sat. They frequently work and reach across each other, without hesitation or utterance. The patchwork partitioning coincides with more frequent verbal coordination utterances, suggesting that the freedom to work anywhere in the workspace requires greater coordination effort between the collaborators. These differences suggest that tabletop applications and interaction techniques that are proven in co-located settings might, therefore, not afford similar properties if naïvely transferred to remote settings, and may lead to coordination problems. The widely-used Rotate 'N' Translate interaction technique (Kruger et al., 2005), for example, aimed to enable coordination through the orientation of task artefacts (Section 2.2.2). However, if the remote collaborators are overlaid then task-artefact rotation is questionable as a coordination mechanism. Designers of remote tabletop interfaces should therefore be aware of these issues when adopting applications and interaction techniques designed for co-located collaboration.

If remote tabletop collaborators sit in the overlaid arrangement and do not exhibit the spatial partitioning observed in co-located tabletop collaboration, then why apply tabletop interaction techniques that were intended for co-located collaborators sat in a non-overlaid arrangement? The study showed that remote tabletop interfaces enable collaborators to maintain a high level of workspace awareness using the hypothesised mechanisms described earlier. However, the results also showed that direct input and remote arm representations are sometimes not by themselves sufficient. Some interaction techniques impaired workspace awareness because they were not localised to the vicinity of the interacting hand, or did not follow the direct manipulation principles of immediate, incremental, continuous visual changes in the workspace. This supports the theoretical work in Chapter 3 suggesting that interaction techniques following these two additional guidelines will result in greater workspace awareness. Tabletop interaction techniques using small movable task artefacts fulfil these additional design guidelines.

The study also supports previous findings that high-fidelity remote arm representations enable a range of workspace-oriented remote gestures. It found that the lack of depth cues in the remote representation prohibits the use of three-dimensional trajectory in pointing gestures and consequently necessitates leaning by collaborators who wish to gesture to the far side of the workspace.

7.2 Limitations and Future Work

This section considers each contribution in turn and discusses both their limitations as they extend beyond the immediate setting of this dissertation, and also the new research opportunities identified.

7.2.1 Design of Remote Tabletop Interfaces

Extension to Other Applications and Interaction Techniques

The design guidelines for awareness in remote tabletop interfaces have been evaluated only in the room layout application, and only for dragging small task artefacts. The movement of the large floor plan and the opening of the menu did not follow the guidelines (Section 6.5.4), and the tabletop applications implemented in Chapter 5 do not begin to explore the guidelines (Section 5.6).

This dissertation has, nevertheless, described how the guidelines might be fulfilled in interaction techniques beyond the dragging of small task artefacts. Chapter 3 discussed how the guidelines might be fulfilled in a variety of techniques for task artefact creation, task artefact removal, mode selection, and menus. It also presented a compromise solution for the dragging of large task artefacts, to which the guidelines cannot be applied easily. Chapter 3 additionally discussed how the guidelines might be fulfilled by alternative interaction techniques in applications beyond room layout. The collaborative web-browsing and document review applications in Chapter 5, for instance, involved splitting large task artefacts into pages to avoid workspace navigation, while maps and spatially-fixed data could be explored using lenses.

The design guidelines can therefore be fulfilled, in theory at least, in a variety of tabletop applications and interaction techniques. Further work is necessary to explore this in practice to identify the tradeoffs involved, the limiting cases, and how workspace awareness varies among applications and techniques.

Extension to Other Collaborative Technologies

Some of the design guidelines might also be applied to increase workspace awareness in other collaborative technologies. Tabletop interaction techniques such as piling might be employed in conventional groupware systems, for instance, to enable collaborators to work side by side without requiring workspace navigation. This is likely to increase workspace awareness (Section 3.1.3). Translucent fake "arms" could also extend from one side of the workspace to follow the mouse telepointer in systems that avoid workspace navigation. These "arms" might more effectively secure a remote partner's attention during intentional gesture, and also convey greater consequential communication. Conventional groupware systems could also support rich indirect arm and whole-hand gesturing on a small screen by abandoning the mouse in favour of a desktop arm-capture system such as Visual Touchpad (Malik and Laszlo, 2004). The consumer multi-touch monitors currently under development (e.g. Hodges et al., 2007) would additionally enable direct hand input and, perhaps, arm capture for gesturing . In the absence of large horizontal displays in the workplace, techniques to make conventional groupware systems more usable are likely to have a greater immediate impact on people's ability to collaborate remotely. The design guidelines might also increase awareness in other remote collaboration technologies, such as wall displays and augmented remote tangible systems, and also technologies for co-located collaboration, such as tabletop interfaces.

7.2.2 Construction of Remote Tabletop Interfaces

Extension to Other Applications and Interaction Techniques

The method for the rapid construction of high-resolution tabletop applications presents new opportunities to investigate novel tabletop interfaces both for co-located and remote collaboration. I have demonstrated that the method can be applied to create a range of applications.

However, the feasibility of implementing a given application or interaction technique using the method depends on the suitability of the abstractions and the availability of appropriate existing user interface components, and so the method is not without limitations. Chapter 5 described how, for example, a tabletop video editing application, and a fish-eye lens interaction technique, are not suited to the current abstractions. Nevertheless, the method provides a sound basis from which to work in such cases, and the characterisation of its performance and limitations in Chapter 5 enables informed judgements of suitability to be made.

Extension to Other Collaborative Technologies

The method could also be applied to the other collaboration technologies proposed in the previous section. The system already supports mouse and keyboard interaction using a desktop PC, and accordingly could be used to investigate tabletop techniques in conventional groupware. Guimbretière et al. (2001) demonstrated a high-resolution wall display using movable task artefacts for co-located collaboration, and the method could extend this to remote collaboration. The method could also be adapted to provide a limited platform for augmented reality surfaces by updating tile positions based on tracked fiducial markers.

Extension to Future Technology

The chosen architecture, and the bottlenecks identified in Chapter 5, depended upon the capabilities of the present technologies for display, rendering and networking. The movable tile abstraction, for instance, was introduced to avoid bandwidth limitations when moving entities on the display. Multi-projector displays were adopted because of the current limitations of individual projectors and other display technologies.

Improvements in technology in the long-term will necessitate changes to the architecture. In particular, as higher-resolution projectors and panel displays become widely available, the multi-projector approach seems likely to become redundant. Flexible plastic displays are also likely to have a profound impact on this area.

Arm Segmentation for Front-Projected Displays

Section 4.5 described the problems encountered when using image-processing techniques to segment an arm from a camera image of a front-projected display.

The background, from which the arm is to be segmented, changes with the contents of the display, and the projected light discolours the hand. Thermoscopic cameras can be used to address this problem, but are expensive. I adopted a pragmatic approach using a commodity web-cam, and avoided the problems by constraining the blue colour channel of the display. This approach is unlikely to be acceptable outside of the laboratory, however, and the resulting shadow representation does not convey the full detail of the actual hand. More advanced computer vision techniques may offer a more satisfactory solution. A recent algorithm by Coldefy and dit Picard (2007) offers some progress although, as described in Section 4.5, the results are not reliable unless the displayed image is constrained, and it is unclear how the algorithm scales. Future improvements to this algorithm may offer progress in this area.

High-Contrast Features on Multi-Projector Displays

Section 5.2 showed how bilinear interpolation used in geometric correction adversely affected the legibility of text. I briefly described a novel technique by Hereld and Stevens (2005) that is designed to address this problem. Early experiments showed that the technique cannot be applied reliably for the kinds of transformations encountered in multi-projector displays, but that this may be addressed by future improvements.

Real-world Applications

Perhaps most importantly of all, this work aids the feasibility of constructing real-world tabletop applications. The range of applications previously demonstrated using co-located tabletop interfaces has been rather limited, and most work on remote tabletop interfaces has focused on the underlying technology. The ability to implement complex high-resolution interfaces rapidly for remote and co-located collaboration enables a wider range of applications, and faster design iteration, than was previously considered feasible. Future work may investigate applications to address real-world problems using contextual design methods and field studies.

7.2.3 Work Practices in Remote Tabletop Collaboration

Extension to Other Applications

The extension of the findings beyond design tasks depends on a number of factors, and is a potential area of investigation for future work. Seating preference in a text-based task such as document review, for instance, may be largely influenced by the orientation of the text. A shared document might lead collaborators to favour the overlaid arrangement to provide a common orientation with respect to the text, whereas individual copies might lead to more varied preferences. Similarly, further investigation is required to establish the role of spatial partitioning in tasks where there is a large single shared task artefact like a text document. Time-sensitive collaborative map-based tasks such as the UAV-control application (Section 5.6.4) require efficient team coordination and may favour stricter partitioning of space to enable delegation of regions of the map. This may result in a preference for non-overlaid seating arrangements around the table to minimise each collaborator's reach into their own delegated region of responsibility. Tasks in which collaborators take on different roles, or have different degrees of power, may also favour non-overlaid arrangements. In the UAV task, for instance, a team leader may require a small area for planning, in addition to the shared map view. A non-overlaid arrangement enables the planning area to be situated close to the leader without affecting the other collaborators.

Extension to Larger Groups

It is unclear how these findings will extend to larger groups, for instance with 3 or 4 connected remote tabletops. It seems unlikely that a large group can collaborate effectively when all collaborators are mutually overlaid because, as the group grows, it will become more important for each collaborator to reserve resources for themselves, yet more difficult to find space to do so. This may lead to a preference for non-overlaid arrangements and greater spatial partitioning.

Further work is also required to establish how these findings extend to larger groups when some of the collaborators are co-located around the same table. The findings suggest that collaborators may be territorial with co-located partners but not with remote partners. Other work investigating group-to-group collaboration has established how imbalances in workspace awareness (Tang, A., et al., 2005) and the absence of visual cues, such as eye gaze, for establishing trust (Bos et al., 2004; Nguyen and Canny, 2005, 2007) can lead to users favouring interactions with their co-located collaborators over remote collaborators. It is not clear whether the awareness cues arising from the proposed design guidelines would overcome this awareness imbalance in practice.

Comparison to Other Collaborative Technologies

This investigation has not studied empirically how remote tabletop interfaces compare in practice to other interfaces for remote collaboration, such as wall displays and conventional groupware systems. A qualitative comparison, in the style of Rogers and Lindley (2004), might identify tradeoffs between the interfaces for different tasks, and opportunities for combining techniques from different interfaces.

Investigation of Spatial Partitioning

Remote tabletop interfaces may also necessitate their own interaction techniques. Collaborators using remote tabletop interfaces may not be territorial in the conventional sense, but nevertheless partition the space according to visual boundaries. Providing flexible visible boundaries appropriate to the task may therefore aid collaborators in establishing personal space on the surface. Providing each collaborator with a movable coloured palette region onto which task artefacts could be placed might aid collaborators in reserving task artefacts for themselves. In tasks where predetermined boundaries are not appropriate, participants might instead sketch and amend their own visible boundaries.

The absence of territorial partitioning of space observed in the remote tabletop study conditions did not seem to disadvantage the collaborators. Although it coincided with more frequent coordination utterances, all pairs were able to produce satisfactory designs. Future work using an alternative task, in which an absence of territoriality is particularly disadvantageous, might enable more detailed findings about how and why partitioning arises, and the kinds of tasks for which it is advantageous or problematic. The map-based UAV task (Section 5.6.4), for instance, might provide quantitative indicators of coordination effort by measuring group performance at different stress levels. Such studies would also enable empirical investigation of the proposed interventions to aid partitioning, such as sketching visible boundaries.

Remote Gesture

Further work might also address the shortcomings of the remote gesture representation. The system did not convey the three-dimensional trajectory pointing gestures observed in co-located collaboration, and so remote collaborators were required to stretch to gesture to the far side of the table. Effective remote three-dimensional gestures may require a three-dimensional display (Grossman and Wigdor, 2007), for instance using polarising glasses, on which to reconstruct the arm. Alternatively, computer vision techniques could capture the three-dimensional gesture and create a more effective two-dimensional remote representation. The shadow of the hand and arm could be stretched, for instance, so that the end of the outstretched finger correctly reaches the intended referent. This work would build on prior work in reaching techniques for large displays (e.g. Pinelle et al., 2008).

Combining Person and Task Space

Lastly, having characterised the utility of remote tabletop interfaces for collaboration in task-space, future work might begin to consider how to combine it with a person-space technology (defined in Section 2.5 above), such as the MultiView system described in Section 2.6 (Nguyen and Canny, 2005, 2007). Addition of a person-space technology may offer benefits in tasks, such as negotiation tasks, that rely on social cues, as discussed in Section 2.6. However, the person-space technology must faithfully convey spatial cues such as eye gaze, body orientation, and arm location, to avoid impairing the accurate spatial cues conveyed by the remote tabletop task-space technology. There is likely little merit in using spatially-faithful remote gesture techniques in the remote tabletop interface if the person-space technology presents conflicting cues. Some systems, for instance, use an overlaid remote tabletop arrangement yet present a video view of the remote collaborator on a vertical display on the far side of the table (e.g. Izadi et al., 2007). Such systems present conflicting spatial cues because a user sees the remote arm reaching away from them on the table but towards them in the video view.

7.3 Concluding Remarks

I have shown, through the application of theory, and observations of their use in practice, that remote tabletop interfaces are a promising way to support collaborative activity in a shared visual workspace. Furthermore, the previous limitations of display resolution can be overcome to enable rapid exploration of new and potentially-useful remote tabletop applications. However, there is still much progress to be made, in display technologies, in application design, and in the study of work practices, before this technology can be considered useful and feasible on a large scale.

In order to aid further exploration in this area, I have packaged the T3 implementation as a research toolkit which, with the kind agreement of the sponsors of the work, is freely available for non-commercial research. The implementation is available from `http://www.cl.cam.ac.uk/t3/`.

Appendix A

T3 Protocol Entities and Messages

Message Type	**Fields**
Keyboard Input	Client numeric identifier Person numeric identifier Keyboard numeric identifier Message type (numeric field indicating pressed, released or typed) Key code numeric identifier Key modifiers (shift, alt, ctrl, etc. as a 32 bit bitmap) Key character
Point Device Input	Client numeric identifier Person numeric identifier Point input device type (numeric field) Whether location and buttons are currently known (Boolean) Location in surface coordinates Buttons (32 bit bitmap) Extra (Any Serializable object)
Outline Device Input	Client numeric identifier Outline input device type (numeric field) Outline contour as an encoded list of lists of surface coordinates.

Table A.1: Input messages sent from client to the server, and their fields.

Entity	Fields	Operations

Tile	Numeric identifier
	Image width and height in pixels (before scaling or rotating)
	Position of centre in surface coordinates
	Desired width and height in surface coordinates (determines scale)
	Angle of rotation
	Visibility (Boolean)
	Pixel data as a raster
	Order of rendering
	Flags (for future use)
	Type (for future use)
		CreateTile(tileId, tileImageWidth, tileImageHeight, flags, type)
		SetTileVisibility(tileId, visibility)
		SetTileTransform(tileId, surfaceX, surfaceY, surfaceWidth, surfaceHeight, angle)
		UpdatePartOfTileImage(tileId, tileImageX, tileImageY, bufferWidth, bufferHeight, bufferOfPixelData)
		DestroyTile (tileId)
		SetTileAndLinkOrder(listOfTileIdsAndLinkIdsFromFro ntToBack)

Link	Numeric identifier Colour Rectangle A, specified as a scale, rotation and translation to be applied to the unit square positioned at the origin Rectangle B, specified similarly Flags (for future use) Type (for future use)	CreateLink(linkId, flags, type, colour, transformationOfUnitSquareAtOriginToRectangleA, transformationOfUnitSquareAtOriginToRectangleB) SetLinkTransforms(linkId, flags, type, colour, transformationOfUnitSquareAtOriginToRectangleA, transformationOfUnitSquareAtOriginToRectangleB) DestroyLink(linkId)

Cursor	Numeric identifier
	Type (outline or trail)
	Colour
	List of trail points, where each point has surface coordinates, timestamp and Boolean visibility.
	Outline contour as an encoded list of lists of surface coordinates.
	Flags (for future use)
		CreateCursor(cursorId, colour, flags, type)
		AddNewCursorTrailPoint(cursorId, surfaceX, surfaceY, visibility)
		SetCursorOutline(cursorId, encodedListOfListsOfSurfaceCoordinates)
		Destroy(cursorId)

Table A.2: The entities used in the T3 protocol, their fields, and their operations.

Appendix B

Locking in T3 and Swing

Implementing the modified Swing repaint manager requires care to avoid deadlock. Its methods may be called simultaneously by different threads, and so a lock, X, is needed to protect the data structure of stored dirty region coordinates. During painting, this data structure must be accessed but X cannot be held without risking deadlock, for the following reasons. Suppose a Swing component has its own lock, Y, to protect its shared data structures. Now, a repainting thread is holding X while repainting, and waiting on Y so that it can paint the Swing component. Another thread may be holding Y and blocked awaiting X so that it can add another dirty region, causing deadlock.

This presents a problem because, during painting, the data structure must be accessed but its lock cannot be held. Making a copy of the data structure before starting to repaint avoids the problem, but is time-consuming for such a regular operation. T3 solves this problem by using two data structures, one used while repainting and one while adding dirty regions. The data structures are swapped just before repainting begins. The lock X is held only while swapping, not while repainting, so there is no risk of deadlock.

Care is also required when integrating the T3 and Swing synchronization systems. According to Swing conventions, off-screen Swing rendering for T3 must be conducted by one particular thread, the Swing thread. During this process, the Swing thread must acquire the T3 lock, described previously, in order to update the state manager. This may again cause deadlock because T3 lock may be held by another thread that is blocked waiting for the Swing thread to finish repainting and to carry out some other operation. The JEditorPane HTML renderer class is a component that often blocks awaiting the Swing thread to execute arbitrary code.

This second problem is avoided by ensuring that application programmers do not manipulate Swing components while holding the T3 lock. Instead, such manipulations must be performed asynchronously without holding the T3 lock, by passing them as a Runnable object to Swing's invokeLater method.

Appendix C

Experiment Materials

C.1 Task Briefs

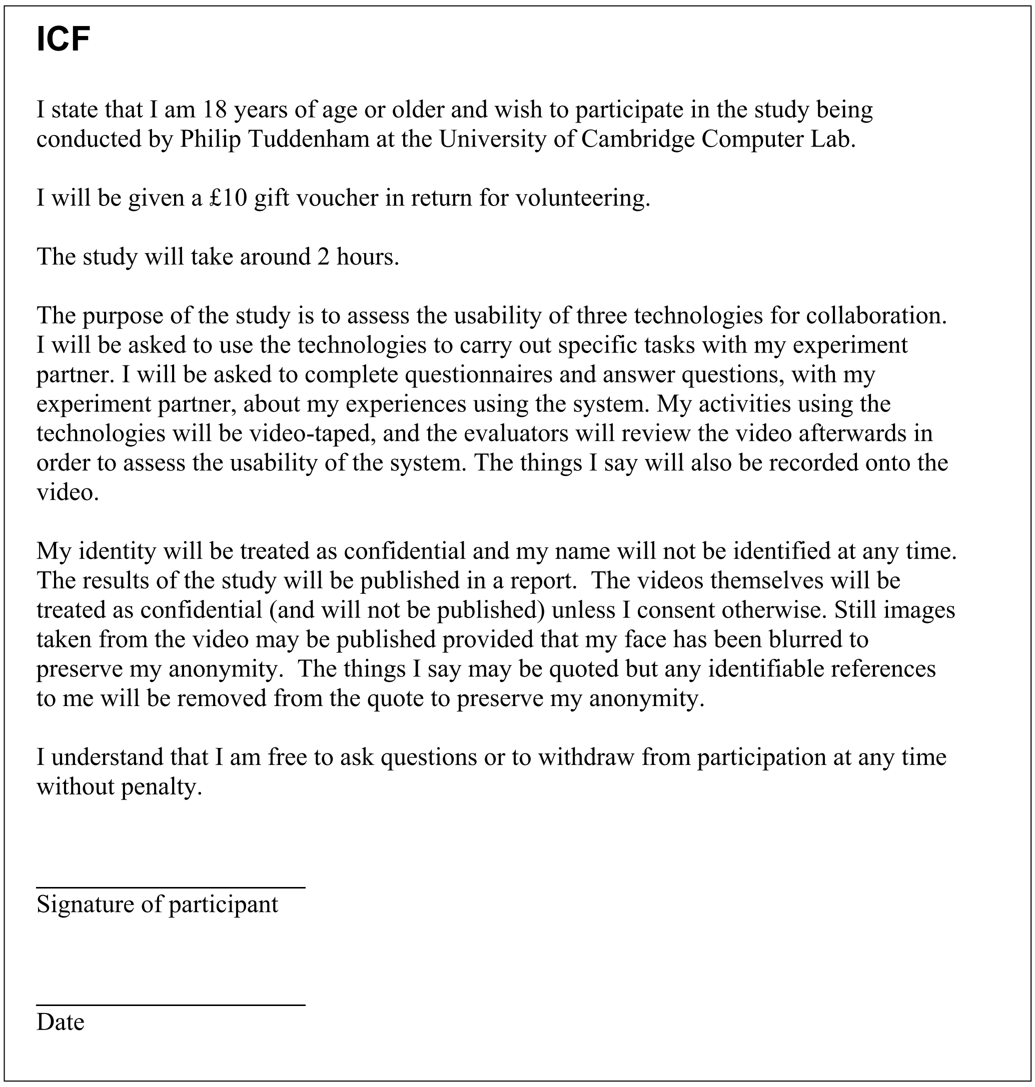
Practice Task. Design a new coffee shop and allow space for: a small kitchen; a rack of newspapers; the counter itself; individuals to read; groups to sit and chat; storing supplies; Seat as many people as poss.

Task 1. Design a new space for a University research group and allow space for: individual work; group work; informal gatherings; a professor, a secretary, research assistants, a visiting academic, PhD students, and interns; journals and books; equipment; as many desks as possible.

Task 2. Design a new reading area for a University library and allow space for: new books and journals; groups to work; individuals to work; photocopying; computer catalogue; librarians; as many tables for readers as possible.

Task 3. Design a new communal space for a University halls of residence and allow space for: watching TV; playing games - pool, xbox, board games; quiet reading and discussions; serving drinks; pigeon holes; student union committee; as many people to use the area as possible.

C.2 Informed Consent Form



ICF

I state that I am 18 years of age or older and wish to participate in the study being conducted by Philip Tuddenham at the University of Cambridge Computer Lab.

I will be given a £10 gift voucher in return for volunteering.

The study will take around 2 hours.

The purpose of the study is to assess the usability of three technologies for collaboration. I will be asked to use the technologies to carry out specific tasks with my experiment partner. I will be asked to complete questionnaires and answer questions, with my experiment partner, about my experiences using the system. My activities using the technologies will be video-taped, and the evaluators will review the video afterwards in order to assess the usability of the system. The things I say will also be recorded onto the video.

My identity will be treated as confidential and my name will not be identified at any time. The results of the study will be published in a report. The videos themselves will be treated as confidential (and will not be published) unless I consent otherwise. Still images taken from the video may be published provided that my face has been blurred to preserve my anonymity. The things I say may be quoted but any identifiable references to me will be removed from the quote to preserve my anonymity.

I understand that I am free to ask questions or to withdraw from participation at any time without penalty.

____
Signature of participant

____
Date

(I gained additional consent in order to use unblurred video frames in this dissertation.)

C.3 Pre-study Questionnaire

Pre-study questionnaire

Participant number:

Name:

E-mail address:

Age (please circle one):

Under 20	20-29	30-39	40-49	Over 49

Gender (please circle one):

Male	Female

Do you know your experiment partner (please circle one)?

Never met	Met each other before	Know each other well

Handedness (please circle one):

Left-handed	Right-handed

Do you have prior experience using interactive tabletop computers?

C.4 Post-condition Questionnaire

End-of-technology questionnaire

Participant number:

Technology:

For each of the following statements, please circle the number that most closely corresponds to your opinion.

I thought that our final layout matched the requirements very well.
(1 represents strongly agree, and 7 represents strongly disagree).

1 2 3 4 5 6 7

I was always aware that my partner and I were in different rooms.
(1 represents strongly agree, and 7 represents strongly disagree).

1 2 3 4 5 6 7

I was always aware of my partner's presence.
(1 represents strongly agree, and 7 represents strongly disagree).

1 2 3 4 5 6 7

It felt as if we were sitting at the same table.
(1 represents strongly agree, and 7 represents strongly disagree).

1 2 3 4 5 6 7

How easy or hard was the task to complete using this technology?
(1 represents very easy, and 7 represents very hard).

1 2 3 4 5 6 7

How did you find communicating this way?
(1 represents very easy, and 7 represents very hard).

1 2 3 4 5 6 7

How easily did you understand what your partner was saying?
(1 represents very easy to understand, and 7 represents very hard to understand).

1 2 3 4 5 6 7

I was confident that my partner was aware of what I was doing at all times.
(1 represents strongly agree, and 7 represents strongly disagree).

1 2 3 4 5 6 7

I was aware of what my partner was doing at all times.
(1 represents strongly agree, and 7 represents strongly disagree).

1 2 3 4 5 6 7

We worked together throughout the task.
(1 represents strongly agree, and 7 represents strongly disagree).

1 2 3 4 5 6 7

Did you have any difficulties using the technology?

Did you think the seating arrangement was appropriate? How did you feel about it?

Was there anything that you wish you could change about the technology?

C.5 Post-study Questionnaire

End-of-study questionnaire

Participant number:

Which technology did you prefer when you were sat in different rooms (please circle one)?

Sat in different rooms, other person's hands appeared immediately in front of me.

Sat in different rooms, other person's hands appeared to one side of me.

Don't know

Do you have any comments as to why you preferred that technology?

Do you have any further comments about any of the technologies or the experiment?

Appendix D

Activity Maps

Activity maps are shown overleaf.

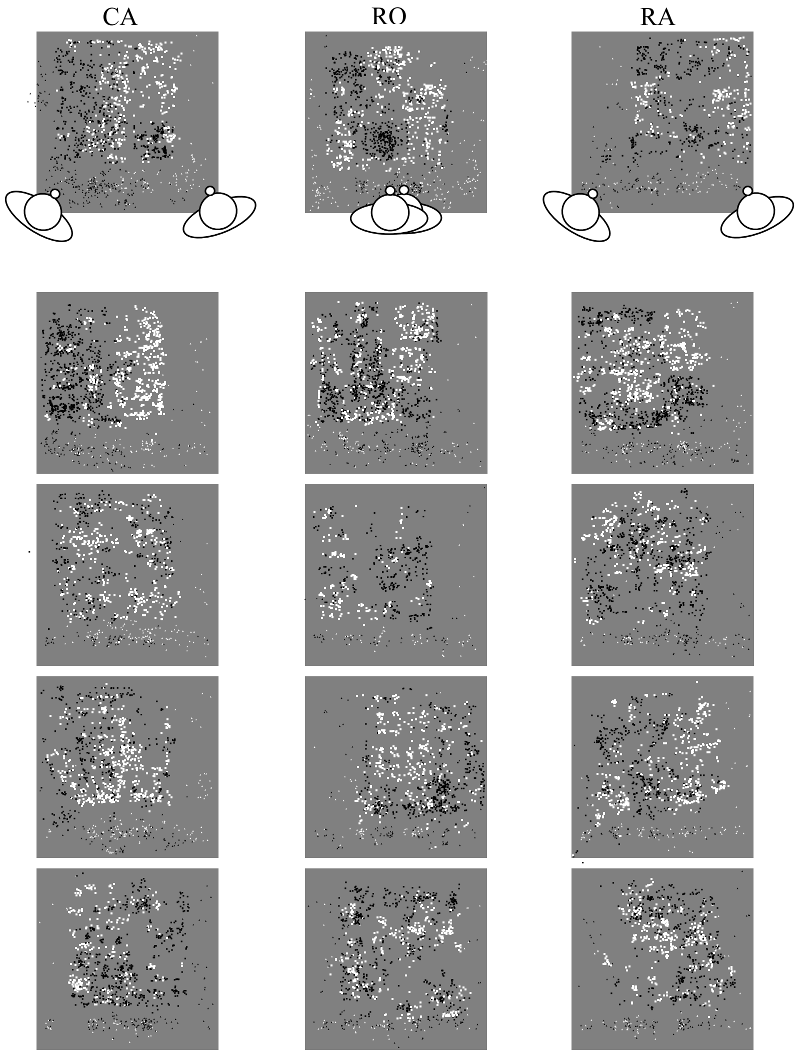


Figure D.1: Activity maps for pairs 1–5.

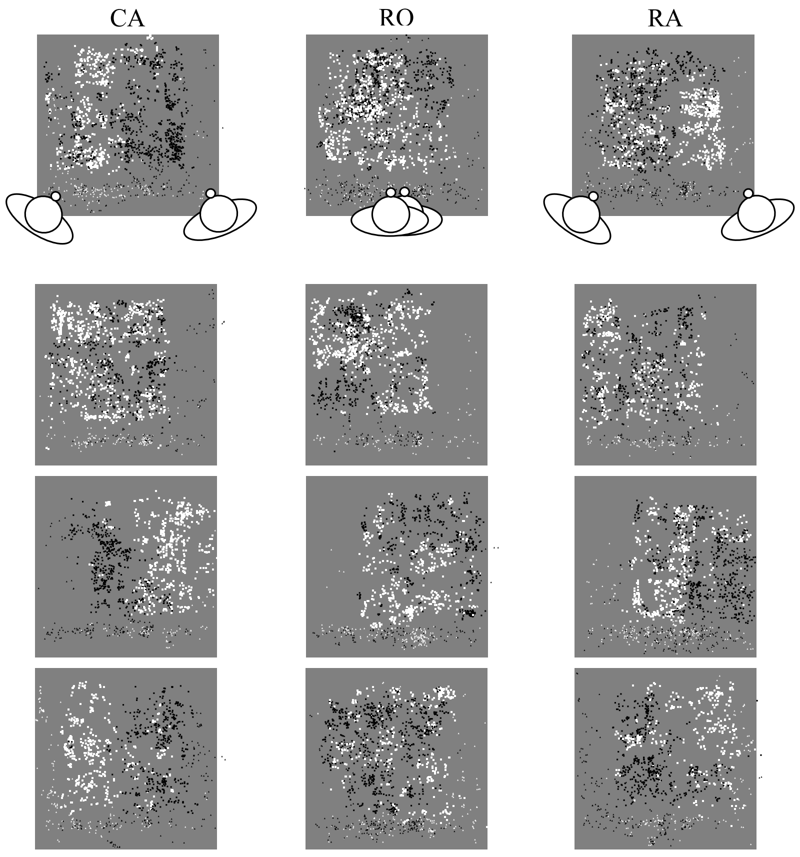


Figure D.2: Activity maps for pairs 6–9.

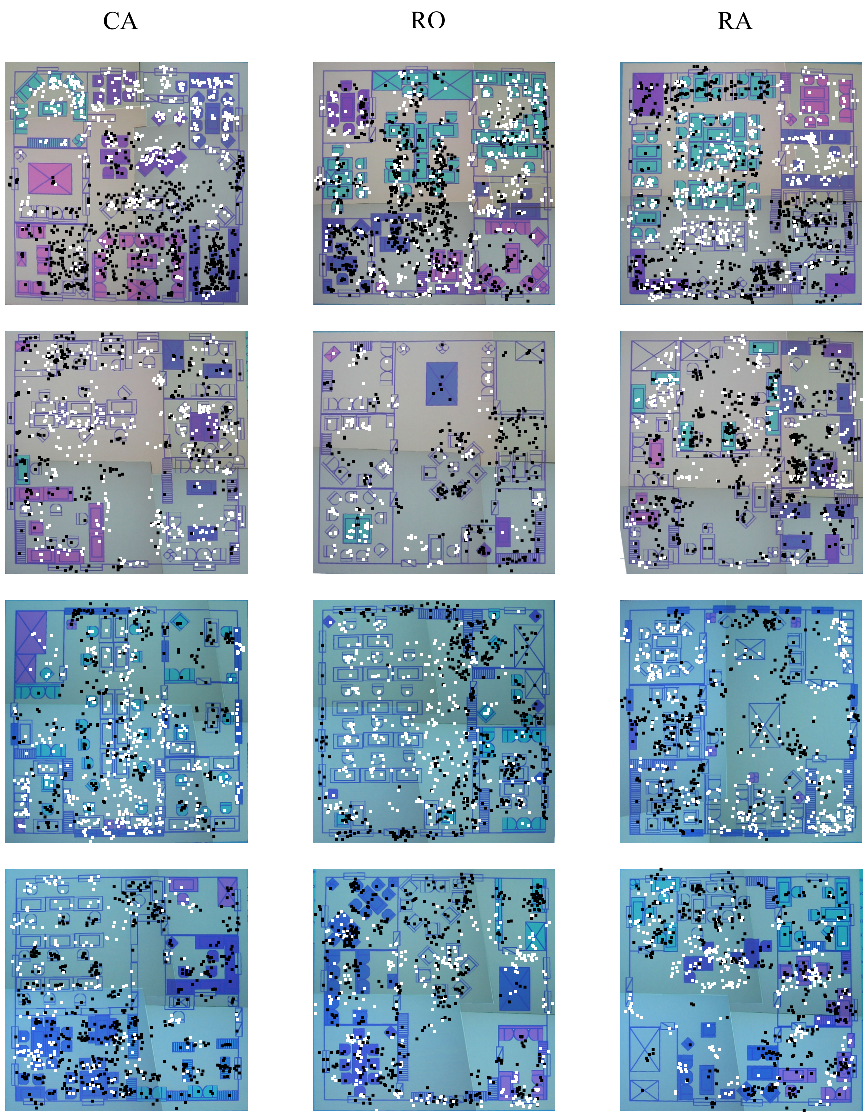


Figure D.3: Interactions with furniture on the floor plan overlaid on the participants final outcome, for pairs 2–5. The photo showing the final outcome for pair 1 is missing.

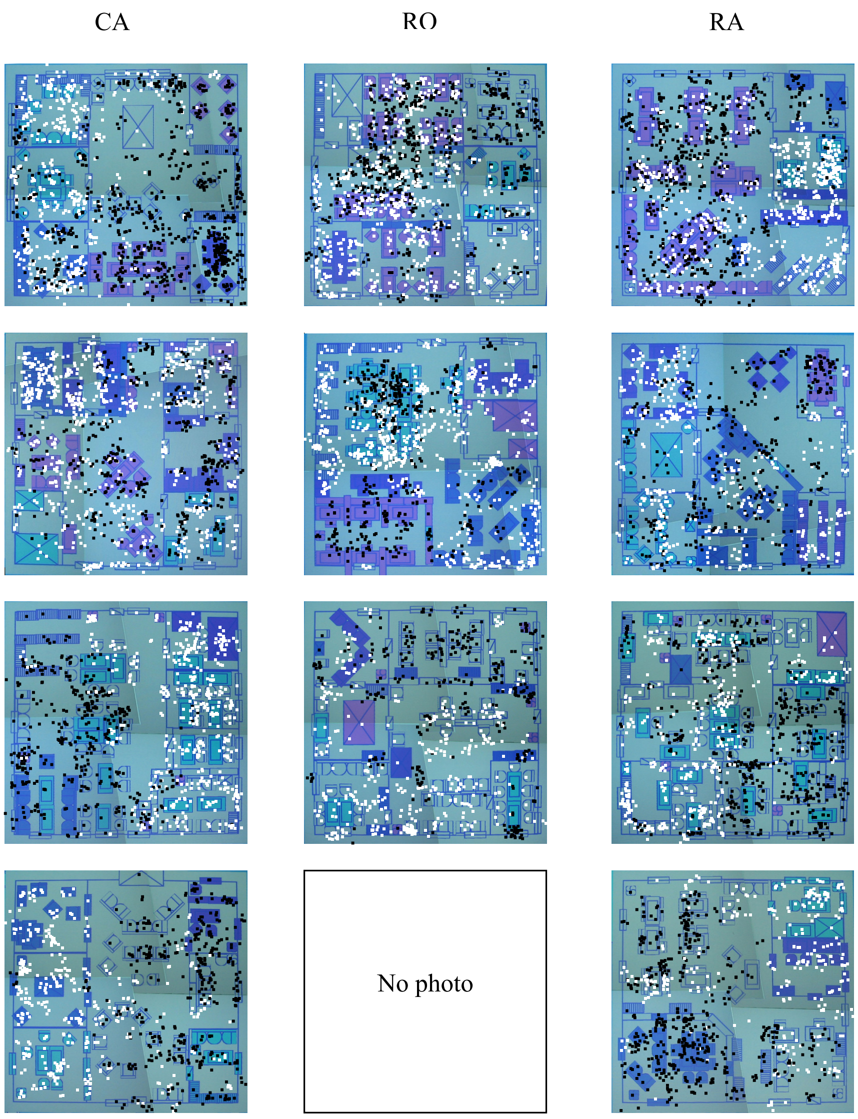


Figure D.4: Interactions with furniture on the floor plan overlaid on the participants final outcome, for pairs 6–9.

Bibliography

Agarwal, A., Izadi, S., Chandraker, M., and Blake, A. (2007). High precision multi-touch sensing on surfaces using overhead cameras. In *Proc. TABLETOP'07: Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems, 2007.*, pages 197–200. 42

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. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 327–332. ACM Press/Addison-Wesley Publishing Co. 31

Amershi, S. and Morris, M. R. (2008). Cosearch: a system for co-located collaborative web search. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1647–1656. ACM Press. 133, 134

Apperley, M., McLeod, L., Masoodian, M., Paine, L., Phillips, M., Rogers, B., and Thomson, K. (2003). Use of video shadow for small group interaction awareness on a large interactive display surface. In *AUIC '03: Proceedings of the Fourth Australasian user interface conference on User interfaces 2003*, pages 81–90, Darlinghurst, Australia, Australia. Australian Computer Society, Inc. 174

Apted, T., Kay, J., and Assad, M. (2005). Sharing digital media on collaborative tables and displays. In *Proceedings of The Spaces In-between: Seamful vs. Seamless Interactions (in conjunction with UbiComp'05).* 31, 35, 67, 109, 110

Apted, T., Kay, J., and Quigley, A. (2006). Tabletop sharing of digital photographs for the elderly. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 781–790. ACM Press. 32, 62, 63, 73

Arias, E., Eden, H., and Fisher, G. (1997). Enhancing communication, facilitating shared understanding, and creating better artifacts by integrating physical and computational media for design. In *DIS '97: Proceedings of the 2nd conference on Designing interactive systems*, pages 1–12. ACM Press. 30

Ashdown, M. (2004). *Personal Projected Displays*. PhD thesis, University of Cambridge Computer Laboratory. 34, 40, 41, 49, 67, 76, 82, 85, 89, 90, 93, 97, 104, 105, 109, 110, 137, 150

Ashdown, M. and Cummings, M. (2007). Asymmetric synchronous collaboration within distributed teams. In *7th International Conference on Engineering Psychology and Cognitive Ergonomics at HCII 2007*, pages 245–255. Springer Lecture Notes in Artifical Intellgence. 141

Ashdown, M. and Robinson, P. (2005). Escritoire: A personal projected display. *IEEE MultiMedia*, 12(1):34–42. 20, 21, 22, 34, 35, 49, 63, 82, 85, 109, 110, 137

Beck, K. (1999). *Extreme Programming Explained: Embrace Change*. Addison-Wesley. 138

Bekker, M. M., Olson, J. S., and Olson, G. M. (1995). Analysis of gestures in face-to-face design teams provides guidance for how to use groupware in design. In *DIS '95: Proceedings of the 1st conference on Designing interactive systems*, pages 157–166. ACM Press. 45

Bier, E. A. and Freeman, S. (1991). Mmm: a user interface architecture for shared editors on a single screen. In *UIST '91: Proceedings of the 4th annual ACM symposium on User interface software and technology*, pages 79–86. ACM Press. 28

Bier, E. A., Freeman, S., and Pier, K. (1992). Mmm: The multi-device multi-user multi-editor. In *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 645–646. ACM Press. 28

Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T. D. (1993). Toolglass and magic lenses: the see-through interface. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 73–80. ACM Press. 36, 63, 64, 66

Bly, S. A. (1988). A use of drawing surfaces in different collaborative settings. In *CSCW '88: Proceedings of the 1988 ACM conference on Computer-supported cooperative work*, pages 250–256. ACM Press. 45

Bly, S. A., Harrison, S. R., and Irwin, S. (1993). Media spaces: bringing people together in a video, audio, and computing environment. *Commun. ACM*, 36(1):28–46. 19, 44

Bly, S. A. and Minneman, S. L. (1990). Commune: a shared drawing surface. *SIGOIS Bull.*, 11(2-3):184–192. 46

Bos, N., Shami, N. S., Olson, J. S., Cheshin, A., and Nan, N. (2004). In-group/out-group effects in distributed teams: an experimental simulation. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 429–436. ACM Press. 199

Brave, S., Ishii, H., and Dahley, A. (1998). Tangible interfaces for remote collaboration and communication. In *CSCW '98: Proceedings of the 1998 ACM conference on Computer supported cooperative work*, pages 169–178. ACM Press. 48

Brown, M. S. and Seales, W. B. (2002). A practical and flexible tiled display system. In *PG '02: Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, page 194. IEEE Computer Society. 92

Buxton, W. A. S. (1992). Telepresence: integrating shared task and person spaces. In *Proceedings of the conference on Graphics interface '92*, pages 123–129, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc. 42

Chen, H., Sukthankar, R., Wallace, G., and Li, K. (2002). Scalable alignment of large-format multi-projector displays using camera homography trees. In *VIS '02: Proceedings of the conference on Visualization '02*, pages 339–346. IEEE Computer Society. 93, 117

Chen, M. (2002). Achieving effective floor control with a low-bandwidth gesture-sensitive videoconferencing system. In *Proceedings of the tenth ACM international conference on Multimedia*, pages 476–483. ACM Press. 43

Chen, M. (2003). A low-latency lip-synchronized videoconferencing system. In *Proceedings of the conference on Human factors in computing systems*, pages 465–471. ACM Press. 44

Clark, H. and Brennan, S. (1991). Grounding in communication. In Resnick, L. B., Levine, J. M., and Teasley, S. D., editors, *Perspectives on socially shared cognition*, pages 127–149. American Psychological Association. 52

Coldefy, F. and dit Picard, S. L. (2007). Digitable: an interactive multiuser table for collocated and remote collaboration enabling remote gesture visualization. In *Proc. PROCAMS'07: IEEE Workshop on Projector-Camera Systems*, pages 1–8. 20, 21, 22, 50, 83, 101, 109, 110, 143, 198

Cruz-Neira, C., Sandin, D. J., DeFanti, T. A., Kenyon, R. V., and Hart, J. C. (1992). The cave: audio visual experience automatic virtual environment. *Commun. ACM*, 35(6):64–72. 76

Cummings, M. and Mitchell, P. (2007). Operator scheduling strategies in supervisory control of multiple UAVs. *Aerospace Science and Technology*, 11(4):339–348. 140

Dietz, P. and Leigh, D. (2001). Diamondtouch: a multi-user touch technology. In *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*, pages 219–226. ACM Press. 41, 101

Dourish, P. and Bellotti, V. (1992). Awareness and coordination in shared workspaces. In *CSCW '92: Proceedings of the 1992 ACM conference on Computer-supported cooperative work*, pages 107–114. ACM Press. 54

Edge, D. and Blackwell, A. (2006). Correlates of the cognitive dimensions for tangible user interface. *Journal of Visual Languages and Computing*, 17(4):366–394. 30

Egido, C. (1988). Video conferencing as a technology to support group work: a review of its failures. In *Proceedings of the 1988 ACM conference on Computer-supported cooperative work*, pages 13–24. ACM Press. 19, 43

Elrod, S., Bruce, R., Gold, R., Goldberg, D., Halasz, F., Janssen, W., Lee, D., McCall, K., Pedersen, E., Pier, K., Tang, J., and Welch, B. (1992). Liveboard: a large interactive display supporting group meetings, presentations, and remote collaboration. In *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 599–607. ACM Press. 28

Esenther, A. and Ryall, K. (2006). Remotedt: Support for multi-site table collaboration. In *Proc. CollabTech'06: International Conference on Collaboration Technologies (CollabTech)*. 20, 21, 49, 74

Everitt, K. M., Klemmer, S. R., Lee, R., and Landay, J. A. (2003). Two worlds apart: bridging the gap between physical and virtual media for distributed design collaboration. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 553–560. ACM Press. 48

Finn, K., Sellen, A., and Wilbur, S. (1997). *Video-Mediated Communication*. Lawrence Erlbaum Associates, Inc. Mahwah, NJ, USA. 43

Fish, R. S., Kraut, R. E., and Chalfonte, B. L. (1990). The videowindow system in informal communication. In *CSCW '90: Proceedings of the 1990 ACM* *conference on Computer-supported cooperative work*, pages 1–11. ACM Press. 44

Fish, R. S., Kraut, R. E., Root, R. W., and Rice, R. E. (1992). Evaluating video as a technology for informal communication. In *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 37–48. ACM Press. 44

Fitzmaurice, G. W., Ishii, H., and Buxton, W. A. S. (1995). Bricks: laying the foundations for graspable user interfaces. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 442–449. ACM Press/Addison-Wesley Publishing Co. 29, 30

Fjeld, M. and Takatsuka, M., editors (2006). *First IEEE International Workshop on Horizontal Interactive Human-Computer Systems (Tabletop 2006), 5-7 January 2006, Adelaide, Australia*. IEEE, IEEE Computer Society. 31

Forlines, C. and Shen, C. (2005). Dtlens: multi-user tabletop spatial data exploration. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 119–122. ACM Press. 31, 36, 63, 64, 69

Fraser, M. (2000). *Working with Objects in Collaborative Virtual Environments*. PhD thesis, University of Nottingham. 42, 147, 155

Freeman, S. (1994). *An architecture for distributed user interfaces*. PhD thesis, University of Cambridge. 47

Fussell, S. R., Setlock, L. D., Yang, J., Ou, J., Mauer, E., and Kramer, A. D. I. (2004). Gestures over video streams to support remote collaboration on physical tasks. *Human Computer Interaction*, 19:273–309. 53

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. M. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley. 80

Gaver, W., Moran, T., MacLean, A., Lövstrand, L., Dourish, P., Carter, K., and Buxton, W. (1992). Realizing a video environment: Europarc's rave system. In *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 27–35. ACM Press. 44

Gaver, W. W., Sellen, A., Heath, C., and Luff, P. (1993). One is not enough: multiple views in a media space. In *CHI '93: Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems*, pages 335–341. ACM Press. 54, 147

Gergle, D., Kraut, R. E., and Fussell, S. R. (2004). Action as language in a shared visual space. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 487–496. ACM Press. 52, 164

Gergle, D., Kraut, R. E., and Fussell, S. R. (2006). The impact of delayed visual feedback on collaborative performance. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1303–1312. ACM Press. 52

Greenberg, S., Gutwin, C., and Cockburn, A. (1996). Awareness through fisheye views in relaxed-wysiwis groupware. In *GI '96: Proceedings of the conference on Graphics interface '96*, pages 28–38, Toronto, Ont., Canada, Canada. Canadian Information Processing Society. 54

Greenberg, S. and Roseman, M. (1996). Groupweb: a www browser as real time groupware. In *CHI '96: Conference companion on Human factors in computing systems*, pages 271–272. ACM Press. 134

Greenhalgh, C. and Benford, S. (1995). Massive: a collaborative virtual environment for teleconferencing. *ACM Trans. Comput.-Hum. Interact.*, 2(3):239–261. 42

Grossman, T. and Wigdor, D. (2007). Going Deeper: a Taxonomy of 3D on the Tabletop. In *TABLETOP'07: Proc. Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, pages 137–144. 31, 200

Grudin, J. and Poltrock, S. E. (1997). Computer-supported cooperative work and groupware. In Zelkowitz, M., editor, *Advances in Computers Vol. 45*, pages 269–320. Academic Press. 27

Guimbretière, F., Stone, M., and Winograd, T. (2001). Fluid interaction with high-resolution wall-size displays. In *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*, pages 21–30. ACM Press. 29, 35, 67, 76, 197

Gutwin, C., Fedak, C., Watson, M., Dyck, J., and Bell, T. (2006). Improving network efficiency in real-time groupware with general message compression. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 119–128. ACM Press. 128

Gutwin, C. and Greenberg, S. (1998). Design for individuals, design for groups: tradeoffs between power and workspace awareness. In *CSCW '98: Proceedings* *of the 1998 ACM conference on Computer supported cooperative work*, pages 207–216. ACM Press. 56, 61, 65, 66

Gutwin, C. and Greenberg, S. (1999). The effects of workspace awareness support on the usability of real-time distributed groupware. *ACM Trans. Comput.-Hum. Interact.*, 6(3):243–281. 54, 55, 56, 66, 150, 164, 167

Gutwin, C. and Greenberg, S. (2002). A descriptive framework of workspace awareness for real-time groupware. *Comput. Supported Coop. Work*, 11(3):411–446. 13, 19, 54, 59, 60, 66, 67, 69, 70, 187, 192

Gutwin, C., Roseman, M., and Greenberg, S. (1996). A usability study of awareness widgets in a shared workspace groupware system. In *CSCW '96: Proceedings of the 1996 ACM conference on Computer supported cooperative work*, pages 258–267. ACM Press. 54, 166

Hall, E. T. (1966). *Distances in Man: The Hidden Dimension*. DoubleDay, Garden City, NY. 71

Haller, M., Brandl, P., Leithinger, D., Leitner, J., Seifried, T., and Billinghurst, M. (2006). Shared Design Space: Sketching ideas using digital pens and a large augmented tabletop setup. In *ICAT'06: Proc. 16th International Conference on Artificial Reality and Telexistence*, volume 4282, pages 948–959. Springer LNCS. 41

Han, J. Y. (2005). Low-cost multi-touch sensing through frustrated total internal reflection. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 115–118. ACM Press. 41

Han, R., Perret, V., and Naghshineh, M. (2000). Websplitter: a unified xml framework for multi-device collaborative web browsing. In *CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 221–230. ACM Press. 134

Hartley, R. and Zisserman, A. (2000). *Multiple view geometry in computer vision*. Cambridge University Press. 93

Hauber, J., Regenbrecht, H., Billinghurst, M., and Cockburn, A. (2006). Spatiality in videoconferencing: trade-offs between efficiency and social presence. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 413–422. ACM Press. 51, 72, 166

Heath, C., Jirotka, M., Luff, P., and Hindmarsh, J. (1994). Unpacking collaboration: the interactional organisation of trading in a city dealing room. *Computer Supported Cooperative Work (CSCW)*, 3(2):147–165. 146

Heath, C. and Luff, P. (1992). Collaboration and control: crisis management and multimedia technology in London Underground Line Control Rooms. *Computer Supported Cooperative Work*, 1(1):69–94. 54, 146

Heath, C., Luff, P., Kuzuoka, H., Yamazaki, K., and Oyama, S. (2001). Creating coherent environments for collaboration. In *ECSCW'01: Proceedings of the seventh conference on European Conference on Computer Supported Cooperative Work*, pages 119–138, Norwell, MA, USA. Kluwer Academic Publishers. 54

Hereld, M., Judson, I. R., and Stevens, R. (2002). Dottytoto: A measurement engine for aligning multi-projector display systems. Technical Report ANL/MCS-P958-0502, Argonne National Laboratory. 92

Hereld, M. and Stevens, R. (2005). Pixel-aligned warping for multiprojector tiled displays. In *PROCAMS 2005*. 120, 198

Hill, R. D., Brinck, T., Rohall, S. L., Patterson, J. F., and Wilner, W. (1994). The rendezvous architecture and language for constructing multiuser applications. *ACM Trans. Comput.-Hum. Interact.*, 1(2):81–125. 80

Hindmarsh, J., Fraser, M., Heath, C., Benford, S., and Greenhalgh, C. (2000). Object-focused interaction in collaborative virtual environments. *ACM Trans. Comput.-Hum. Interact.*, 7(4):477–509. 42

Hindus, D., Ackerman, M. S., Mainwaring, S., and Starr, B. (1996). Thunderwire: a field study of an audio-only media space. In *CSCW '96: Proceedings of the 1996 ACM conference on Computer supported cooperative work*, pages 238–247. ACM Press. 44

Hinrichs, U., Carpendale, S., Scott, S. D., Pattison, E., Butz, A., Fisher, B., Krger, A., and Olivier, P. (2005). Interface currents: Supporting fluent collaboration on tabletop displays. In *Proc. of Smart Graphics*. 13, 20, 32, 35, 63, 68, 73, 76, 84

Hodges, S., Izadi, S., Butler, A., Rrustemi, A., and Buxton, B. (2007). Thinsight: versatile multi-touch sensing for thin form-factor displays. In *UIST '07: Proceedings of the 20th annual ACM symposium on User interface software and technology*, pages 259–268. ACM Press. 196

Hornecker, E. and Buur, J. (2006). Getting a grip on tangible interaction: a framework on physical space and social interaction. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 437–446, New York, NY, USA. ACM Press. 30

Huang, E., Mynatt, E., and Trimble, J. (2006). Displays in the Wild: Understanding the Dynamics and Evolution of a Display Ecology. *Proceedings of the Fourth International Conference on Pervasive Computing*. 29

Hughes, J., King, V., Rodden, T., and Andersen, H. (1994). Moving out from the control room: ethnography in system design. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 429–439. ACM Press. 146

Humphreys, G. and Hanrahan, P. (1999). A distributed graphics system for large tiled displays. In *Proceedings of the conference on Visualization'99: celebrating ten years*, pages 215–223. IEEE Computer Society Press Los Alamitos, CA, USA. 76

Humphreys, G., Houston, M., Ng, R., Frank, R., Ahern, S., Kirchner, P. D., and Klosowski, J. T. (2002). Chromium: a stream-processing framework for interactive rendering on clusters. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 693–702. ACM Press. 76, 82, 85, 92

Hutterer, P., Close, B. S., and Thomas, B. H. (2006). Supporting mixed presence groupware in tabletop applications. In *TABLETOP '06: Proceedings of the First IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, pages 63–70. IEEE Computer Society. 20, 21, 49, 74, 83, 109, 110

Hutterer, P. and Thomas, B. H. (2007). Groupware support in the windowing system. In *AUIC '07: Proceedings of the eight Australasian conference on User interface*, pages 39–46, Darlinghurst, Australia, Australia. Australian Computer Society, Inc. 109

Isenberg, T., Miede, A., and Carpendale, S. (2006). A buffer framework for supporting responsive interaction in information visualization interfaces. In *Proc. Fourth International Conference on Creating, Connecting and Collaborating through Computing (C5'06)*, pages 262–269. IEEE Computer Society. 32, 73, 76, 125

Ishii, H. (1990). Teamworkstation: towards a seamless shared workspace. In *CSCW '90: Proceedings of the 1990 ACM conference on Computer-supported cooperative work*, pages 13–26. ACM Press. 47

Ishii, H. and Kobayashi, M. (1992). Clearboard: a seamless medium for shared drawing and conversation with eye contact. In *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 525–532. ACM Press. 46, 47

Izadi, S., Agarwal, A., Criminisi, A., Winn, J., Blake, A., and Fitzgibbon, A. (2007). C-slate: A multi-touch and object recognition system for remote collaboration using horizontal surfaces. In *Proc. TABLETOP'07: Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems, 2007.*, pages 3–10. 20, 21, 22, 50, 84, 143, 201

Jakobsen, M. R. and Hornbaek, K. (2006). Evaluating a fisheye view of source code. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 377–386. ACM Press. 138

Jancke, G., Venolia, G. D., Grudin, J., Cadiz, J. J., and Gupta, A. (2001). Linking public spaces: technical and social issues. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 530–537. ACM Press. 44

Jeong, B., Jagodic, R., Renambot, L., Singh, R., Johnson, A., and Leigh, J. (2005). Scalable graphics architecture for high-resolution displays. In *Proc. IEEE Workshop on Information Visualization*. 76, 85

Jordà, S. (2003). Sonigraphical instruments: from fmol to the reactable. In *NIME '03: Proceedings of the 2003 conference on New interfaces for musical expression*, pages 70–76. National University of Singapore. 30

Karahalios, K. and Donath, J. (2004). Telemurals: linking remote spaces with social catalysts. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 615–622. ACM Press. 44

Kilgard, M. (1996). *OpenGL for the X Window System*. Addison-Wesley. 82

Kirk, D. (2007). *Turn It This Way: Remote gesturing in Video-Mediated Communication*. PhD thesis, University of Nottingham. 48, 52, 53, 71, 155, 177

Kirk, D., Crabtree, A., and Rodden, T. (2005). Ways of the hands. In *ECSCW'05: Proceedings of the ninth conference on European Conference on Computer Supported Cooperative Work*, pages 1–21. Springer-Verlag New York, Inc. 53, 60, 187

Kirk, D. and Fraser, D. S. (2006). Comparing remote gesture technologies for supporting collaborative physical tasks. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1191–1200. ACM Press. 53

Kirk, D., Rodden, T., and Fraser, D. S. (2007). Turn it this way: grounding collaborative action with remote gestures. In *CHI '07: Proceedings of the*

SIGCHI conference on Human factors in computing systems, pages 1039–1048. ACM Press. 53, 164

Koike, H., Sato, Y., and Kobayashi, Y. (2001). Integrating paper and digital information on enhanceddesk: a method for realtime finger tracking on an augmented desk system. *ACM Trans. Comput.-Hum. Interact.*, 8(4):307–322. 101

Kraut, R. E., Gergle, D., and Fussell, S. R. (2002). The use of visual information in shared visual spaces: informing the development of virtual co-presence. In *CSCW '02: Proceedings of the 2002 ACM conference on Computer supported cooperative work*, pages 31–40. ACM Press. 52

Krueger, M. W. (1993). Environmental technology: making the real world virtual. *Commun. ACM*, 36(7):36–37. 42

Krueger, M. W., Gionfriddo, T., and Hinrichsen, K. (1985). Videoplace—an artificial reality. *SIGCHI Bull.*, 16(4):35–40. 42

Kruger, R., Carpendale, S., Scott, S. D., and Greenberg, S. (2004). Roles of orientation in tabletop collaboration: Comprehension, coordination and communication. *Comput. Supported Coop. Work*, 13(5-6):501–537. 33, 35, 37, 40, 70, 72, 147

Kruger, R., Carpendale, S., Scott, S. D., and Tang, A. (2005). Fluid integration of rotation and translation. In *Proc. CHI 2005*, pages 601–610. 32, 33, 34, 72, 130, 134, 150, 194

Krumbholz, C., Leigh, J., Johnson, A., Renambot, L., and Kooima, R. (2005). Lambda table: High resolution tiled display table for interacting with large visualizations paper. In *Proc. Workshop on Advanced Collaborative Environments*. 13, 75

Kuzuoka, H., Yamashita, J., Yamazaki, K., and Yamazaki, A. (1999). Agora: a remote collaboration system that enables mutual monitoring. In *CHI '99: CHI '99 extended abstracts on Human factors in computing systems*, pages 190–191. ACM Press. 47

Lauwers, J. C., Joseph, T. A., Lantz, K. A., and Romanow, A. L. (1990). Replicated architectures for shared window systems: a critique. *SIGOIS Bull.*, 11(2-3):249–260. 80, 83, 84

Leithinger, D. and Haller, M. (2007). Improving Menu Interaction for Cluttered Tabletop Setups with User-Drawn Path Menus. In *Proc. TABLETOP'07: Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, pages 121–128. 63, 64

Letessier, J. and Bérard, F. (2004). Visual tracking of bare fingers for interactive surfaces. In *UIST '04: Proceedings of the 17th annual ACM symposium on User interface software and technology*, pages 119–122. ACM Press. 101

Li, K., Chen, H., Chen, Y., Clark, D. W., Cook, P., Damianakis, S., Essl, G., Finkelstein, A., Funkhouser, T., Housel, T., Klein, A., Liu, Z., Praun, E., Samanta, R., Shedd, B., Singh, J. P., Tzanetakis, G., and Zheng, J. (2000). Building and using a scalable display wall system. *IEEE Computer Graphics and Applications*, 20(4):29–37. 76

Loftus, G. R. and Masson, M. E. (1994). Using confidence intervals in within-subject designs. *Psychonomic Bulletin & Review*, 1((4)):476–490. 162

Luff, P., Heath, C., Kuzuoka, H., Yamazaki, K., and Yamashita, J. (2006). Handling documents and discriminating objects in hybrid spaces. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 561–570. ACM Press. 47, 54

Majumder, A. and Brown, M. S. (2007). *Practical Multi-projector Display Design*. A. K. Peters Ltd. 92, 95, 97

Majumder, A. and Stevens, R. (2004). Color nonuniformity in projection-based displays: Analysis and solutions. *IEEE Transactions on Visualization and Computer Graphics*, 10(2):177–188. 95

Malik, S. and Laszlo, J. (2004). Visual touchpad: a two-handed gestural input device. In *ICMI '04: Proceedings of the 6th international conference on Multimodal interfaces*, pages 289–296. ACM Press. 196

Malone, T. W. (1983). How do people organize their desks?: Implications for the design of office information systems. *ACM Trans. Inf. Syst.*, 1(1):99–112. 34

Mantei, M. M., Baecker, R. M., Sellen, A. J., Buxton, W. A. S., Milligan, T., and Wellman, B. (1991). Experiences in the use of a media space. In *CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 203–208. ACM Press. 44

Matsushita, M., Iida, M., Ohguro, T., Shirai, Y., Kakehi, Y., and Naemura, T. (2004). Lumisight table: a face-to-face collaboration support system that optimizes direction of projected information to each stakeholder. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 274–283. ACM Press. 33, 35

Matsushita, N. and Rekimoto, J. (1997). Holowall: designing a finger, hand, body, and object sensitive wall. In *UIST '97: Proceedings of the 10th annual ACM symposium on User interface software and technology*, pages 209–210. ACM Press. 41

Mazalek, A., Reynolds, M., and Davenport, G. (2007). The tviews table in the home. In *Tabletop'07: Second IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, pages 52–59. IEEE Computer Society. 31

McGrath, J. E. (1984a). Methods for the study of groups. In *Groups: Interaction and Performance*, pages 51–66. Prentice-Hall. 147

McGrath, J. E. (1984b). A typology of tasks. In *Groups: Interaction and Performance*, pages 51–66. Prentice-Hall. 150

Miede, A. (2006). Realizing Responsive Interaction for Tabletop Interaction Metaphors. Master's thesis, Otto-von-Guericke-Universität Magdeburg. 76, 125

Miwa, Y. and Ishibiki, C. (2004). Shadow communication: system for embodied interaction with remote partners. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 467–476. ACM Press. 47

Morikawa, O. and Maesako, T. (1998). Hypermirror: toward pleasant-to-use video mediated communication system. In *Proc. CSCW 1998*, pages 149–158. 44

Morris, M. R. (2006). *Supporting Effective Interaction with Tabletop Groupware*. PhD thesis, Stanford University. 20, 31

Morris, M. R., Brush, A. J. B., and Meyers, B. (2007). Reading revisited: Evaluating the usability of digital display surfaces for active reading tasks. In *Tabletop'07: Second IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, pages 79–86. IEEE Computer Society. 134

Morris, M. R. and Horvitz, E. (2007). Searchtogether: an interface for collaborative web search. In *UIST '07: Proceedings of the 20th annual ACM symposium on User interface software and technology*, pages 3–12. ACM Press. 134

Morris, M. R., Huang, A., Paepcke, A., and Winograd, T. (2006a). Cooperative gestures: multi-user gestural interactions for co-located groupware. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1201–1210. ACM Press. 31, 36

Morris, M. R., Morris, D., and Winograd, T. (2004a). Individual audio channels with single display groupware: effects on communication and task strategy. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 242–251. ACM Press. 31

Morris, M. R., Paepcke, A., Winograd, T., and Stamberger, J. (2006b). Teamtag: exploring centralized versus replicated controls for co-located tabletop groupware. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1273–1282. ACM Press. 36, 63, 65

Morris, M. R., Ryall, K., Shen, C., Forlines, C., and Vernier, F. (2004b). Beyond "social protocols": multi-user coordination policies for co-located groupware. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 262–265. ACM Press. 36

Morrison, G. D. (2005). A camera-based input device for large interactive displays. *IEEE Computer Graphics and Applications*, 25(4):52–57. 41

Myers, B. A., Stiel, H., and Gargiulo, R. (1998). Collaboration using multiple pdas connected to a pc. In *CSCW '98: Proceedings of the 1998 ACM conference on Computer supported cooperative work*, pages 285–294. ACM Press. 28

Mynatt, E. D., Igarashi, T., Edwards, W. K., and LaMarca, A. (1999). Flatland: new dimensions in office whiteboards. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 346–353. ACM Press. 29

Nelson, G. (1991). *System Programming with Modula-3*. Prentice Hall. 108

Nguyen, D. and Canny, J. (2005). Multiview: spatially faithful group video conferencing. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 799–808. ACM Press. 19, 44, 199, 201

Nguyen, D. T. and Canny, J. (2007). Multiview: improving trust in group video conferencing through spatial faithfulness. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1465–1474. ACM Press. 19, 44, 199, 201

O'Hara, K. and Sellen, A. (1997). A comparison of reading paper and on-line documents. In *CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 335–342. ACM Press. 137

O'Hara, K. P., Taylor, A., Newman, W., and Sellen, A. J. (2002). Understanding the materiality of writing from multiple sources. *International Journal of Human-Computer Studies*, 56(3):269–305. 137

Okada, K.-I., Maeda, F., Ichikawaa, Y., and Matsushita, Y. (1994). Multiparty videoconferencing at virtual social distance: Majic design. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 385–393. ACM Press. 19, 44

Olson, G. M., Olson, J. S., Carter, M. R., and Storrsten, M. (1992). Small group design meetings: an analysis of collaboration. *Human-Computer Interaction*, 7. 45

Paek, T., Agrawala, M., Basu, S., Drucker, S., Kristjansson, T., Logan, R., Toyama, K., and Wilson, A. (2004). Toward universal mobile interaction for shared displays. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 266–269. ACM Press. 134

Parker, J. K., Mandryk, R. L., and Inkpen, K. M. (2005). Tractorbeam: seamless integration of local and remote pointing for tabletop displays. In *GI '05: Proceedings of Graphics Interface 2005*, pages 33–40. Canadian Human-Computer Communications Society. 41

Patterson, J. F. (1995). A taxonomy of architectures for synchronous groupware applications. *SIGOIS Bull.*, 15(3):27–29. 80

Pauchet, A., Coldefy, F., Lefebvre, L., Picard, S. L. D., Perron, L., Bouguet, A., Collobert, M., Guerin, J., and Corvaisier, D. (2007). Tabletops: Worthwhile experiences of collocated and remote collaboration. In *Proc. TABLETOP'07: Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems, 2007.*, pages 27–34. 51

Pedersen, E. R., McCall, K., Moran, T. P., and Halasz, F. G. (1993). Tivoli: an electronic whiteboard for informal workgroup meetings. In *CHI '93: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 391–398. 28, 29

Pferd, W., Peralta, L., and Prendergast, F. (1979). Special Feature: Interactive Graphics Teleconferencing. *Computer*, 12(11):62–72. 44

Phillips, W. (1999). Architectures for synchronous groupware. Technical Report 1999-425, Department of Computing and Information Science, Queen's University. 80, 82

Pinelle, D., Gutwin, C., and Nacenta, M. (2008). The effects of co-present embodiments on awareness and collaboration in tabletop groupware. In *Proceeding of Graphics Interface*. 201

Piper, A. M., O'Brien, E., Morris, M. R., and Winograd, T. (2006). Sides: a cooperative tabletop computer game for social skills development. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 1–10. ACM Press. 31

Raskar, R., Brown, M. S., Yang, R., Chen, W.-C., Welch, G., Towles, H., Seales, B., and Fuchs, H. (1999). Multi-projector displays using camera-based registration. In *VIS '99: Proceedings of the conference on Visualization '99*, pages 161–168. IEEE Computer Society Press. 93, 95

Raskar, R., Welch, G., Cutts, M., Lake, A., Stesin, L., and Fuchs, H. (1998). The office of the future: a unified approach to image-based modeling and spatially immersive displays. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 179–188. ACM Press. 44

Regenbrecht, H., Haller, M., Hauber, J., and Billinghurst, M. (2006). Carpeno: interfacing remote collaborative virtual environments with table-top interaction. *Virtual Reality*, 10(2):95–107. 20, 22, 50

Regenbrecht, H., Wagner, M., and Baratoff, G. (2002). MagicMeeting: A Collaborative Tangible Augmented Reality System. *Virtual Reality*, 6(3):151–166. 42

Rekimoto, J. (1998). A multiple device approach for supporting whiteboard-based interactions. In *CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 344–351. ACM Press/Addison-Wesley Publishing Co. 35

Rekimoto, J. (2002). Smartskin: an infrastructure for freehand manipulation on interactive surfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 113–120. ACM Press. 41

Rekimoto, J. and Saitoh, M. (1999). Augmented surfaces: a spatially continuous work space for hybrid computing environments. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 378–385. ACM Press. 30

Reznik, D. S. and Canny, J. F. (2001). C'mon part, do the local motion! In *Proc ICRA'01: IEEE International Conference on Robotics and Automation*, volume 3, pages 2235–2242. 48

Richardson, T., Stafford-Fraser, Q., Wood, K. R., and Hopper, A. (1998). Virtual network computing. *IEEE Internet Computing*, 02(1):33–38. 74, 82, 85, 89, 109, 110

Ringel, M., Ryall, K., Shen, C., Forlines, C., and Vernier, F. (2004). Release, relocate, reorient, resize: fluid techniques for document sharing on multi-user interactive tables. In *CHI '04: CHI '04 extended abstracts on Human factors in computing systems*, pages 1441–1444. ACM Press. 35

Robinson, J. A. and Robertson, C. (2001). The livepaper system: Augmenting paper on an enhanced tabletop. *COMPUT GRAPHICS (PERGAMON)*, 25(5):731–743. 47

Rogers, Y., Hazlewood, W., Blevis, E., and Lim, Y. (2004). Finger talk: collaborative decision-making using talk and fingertip interaction around a tabletop display. In *CHI '04: CHI '04 extended abstracts on Human factors in computing systems*, pages 1271–1274. ACM Press. 37, 40

Rogers, Y., Lim, Y., and Hazlewood., W. R. (2007). Equal opportunities: Do shareable interfaces promote more group participation than single user displays? Submitted to HCI Journal. 38

Rogers, Y. and Lindley, S. (2004). Collaborating around vertical and horizontal large interactive displays: which way is best? *Interacting with Computers*, 16(6):1133–1152. 20, 37, 38, 200

Roseman, M. and Greenberg, S. (1996). Building real-time groupware with groupkit, a groupware toolkit. *ACM Trans. Comput.-Hum. Interact.*, 3(1):66–106. 44, 74, 84

Ryall, K., Forlines, C., Shen, C., and Morris, M. R. (2004). Exploring the effects of group size and table size on interactions with tabletop shared-display groupware. In *Proc. CSCW 2004*, pages 284–293. 32, 37, 39, 156

Ryall, K. and Scott, S., editors (2007). *Second IEEE International Workshop on Horizontal Interactive Human-Computer Systems (Tabletop 2007), October 10-12 2007, Newport, Rhode Island, USA*. IEEE, IEEE Computer Society. 31

Rymaszewski, M., Au, W., Wallace, M., Winters, C., Ondrejka, C., Batstone-Cunningham, B., and Rosedale, P. (2006). *Second Life: The Official Guide*. SYBEX Inc. Alameda, CA, USA. 42

Salvador, T., Scholtz, J., and Larson, J. (1996). The denver model for groupware design (yeeeeee haaaaaa!). *SIGCHI Bull.*, 28(1):52–58. 37

Scheifler, R. W. and Gettys, J. (1996). *X Window System: Core and extension protocols: X version 11, releases 6 and 6.1*. Digital Press. 82, 85

Scott, S., Wan, J., Rico, A., Furusho, C., and Cummings, M. L. (2007). Aiding team supervision in command and control operations with large-screen displays. In *HSIS 2007: ASNE Human Systems Integration Symposium*, pages 19–21. 140

Scott, S. D. (2005). *Territoriality in Collaborative Tabletop Workspaces*. PhD thesis, University of Calgary, Calgary, Alberta, Canada. 20, 31, 32, 35, 147, 156

Scott, S. D., Carpendale, M. S. T., and Habelski, S. (2005). Storage bins: Mobile storage for collaborative tabletop displays. *IEEE Computer Graphics and Applications*, 25(4):58–65. 20, 31, 32, 34, 35, 36, 37, 38, 39, 62, 63, 67, 70, 150

Scott, S. D., Carpendale, M. S. T., and Inkpen, K. M. (2004). Territoriality in collaborative tabletop workspaces. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 294–303. ACM Press. 34, 35, 37, 38, 70, 71, 150, 156, 161, 167, 181

Segal, L. (1994). Effects of checklist interface on non-verbal crew communications. Technical Report Contractor Report 177639, NASA Ames Research Center. 54

Sellen, A. J. (1995). Remote conversations: The effects of mediating talk with technology. *Human-Computer Interaction*, 10(4):401–444. 44

Sellen, A. J. and Harper, R. H. (2003). *The Myth of the Paperless Office*. MIT Press, Cambridge, MA, USA. 135

Shapiro, D. (1994). The limits of ethnography: combining social sciences for cscw. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 417–428. ACM Press. 146

Shen, C., Everitt, K., and Ryall, K. (2003). Ubitable: Impromptu face-to-face collaboration on horizontal interactive surfaces. In *UbiComp 2003: Ubiquitous Computing: 5th International Conference*. Springer. 35

Shen, C., Lesh, N. B., Vernier, F., Forlines, C., and Frost, J. (2002). Sharing and building digital group histories. In *CSCW '02: Proceedings of the 2002 ACM conference on Computer supported cooperative work*, pages 324–333. ACM Press. 30, 31, 32, 62

Shen, C., Vernier, F. D., Forlines, C., and Ringel, M. (2004). Diamondspin: an extensible toolkit for around-the-table interaction. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 167–174. ACM Press. 32, 33, 34, 35, 67, 73, 77, 104, 109, 110, 193

Shneiderman, B. (1983). Direct manipulation: A step beyond programming languages. *Computer*, 16(8):57–69. 61

Shoemaker, G. B. D. and Inkpen, K. M. (2001). Single display privacyware: augmenting public displays with private information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 522–529. ACM Press. 35

Short, J., Williams, E., and Christie, B. (1976). *The social psychology of telecommunications*. London: Wiley. 43

Sonnenwald, D., Maglaughlin, K., and Whitton, M. (2004). Designing to support situation awareness across distances: an example from a scientific collaboratory. *Information Processing and Management*, 40(6):989–1011. 42

Ståhl, O., Wallberg, A., Söderberg, J., Humble, J., Fahlén, L. E., Bullock, A., and Lundberg, J. (2002). Information exploration using the pond. In *CVE '02: Proceedings of the 4th international conference on Collaborative virtual environments*, pages 72–79. ACM Press. 30, 31

Stanton, D., Bayon, V., Neale, H., Ghali, A., Benford, S., Cobb, S., Ingram, R., O'Malley, C., Wilson, J., and Pridmore, T. (2001). Classroom collaboration in the design of tangible interfaces for storytelling. In *CHI '01: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 482–489. ACM Press. 28

Stefik, M., Bobrow, D. G., Foster, G., Lanning, S., and Tatar, D. (1987a). Wysiwis revised: early experiences with multiuser interfaces. *ACM Trans. Inf. Syst.*, 5(2):147–167. 66

Stefik, M., Foster, G., Bobrow, D. G., Kahn, K., Lanning, S., and Suchman, L. (1987b). Beyond the chalkboard: computer support for collaboration and problem solving in meetings. *Commun. ACM*, 30(1):32–47. 28, 66

Stewart, J., Bederson, B. B., and Druin, A. (1999). Single display groupware: a model for co-present collaboration. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 286–293. ACM Press. 28

Stone, M. C. (2001). Color and brightness appearance issues in tiled displays. *IEEE Comput. Graph. Appl.*, 21(5):58–66. 95

Streitz, N. A., Geissler, J., Holmer, T., Konomi, S., Muller-Tomfelde, C., Reischl, W., Rexroth, P., Seitz, P., and Steinmetz, R. (1999). i-land: an interactive landscape for creativity and innovation. In *Proc. CHI 1999*, pages 120–127. 30, 31, 32, 34

Stults, R. (1986). Media Space. Technical report, Xerox Palo Alto Research Centre. 44

Takao, N., Shi, J., and Baker, S. (2003). Tele-graffiti: A camera-projector based remote sketching system with hand-based user interface and automatic session summarization. *Int. J. Comput. Vision*, 53(2):115–133. 47

Tandler, P., Prante, T., Müller-Tomfelde, C., Streitz, N., and Steinmetz, R. (2001). Connectables: dynamic coupling of displays for the flexible creation of shared workspaces. In *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*, pages 11–20. ACM Press. 30, 31, 32, 34

Tang, A., Boyle, M., and Greenberg, S. (2005). Understanding and mitigating display and presence disparity in mixed presence groupware. *Journal of Research and Practice in Information Technology*, 37(2). 199

Tang, A., Neustaedter, C., and Greenberg, S. (2006a). Videoarms: Embodiments for mixed presence groupware. In *Proc. HCI 2006: Proceedings of the 20th British HCI Group Annual Conference*, pages 85–102. 20, 22, 50, 51, 60, 84, 101

Tang, A., Tory, M., Po, B., Neumann, P., and Carpendale, S. (2006b). Collaborative coupling over tabletop displays. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1181–1190. ACM Press. 20, 31, 35, 36, 37, 63, 70, 147, 167, 168, 185, 187, 192

Tang, J. C. (1991). Findings from observational studies of collaborative work. *Int. J. Man-Mach. Stud.*, 34(2):143–160. 35, 36, 37, 38, 39, 40, 45, 70, 71, 150

Tang, J. C. and Minneman, S. (1991a). Videowhiteboard: video shadows to support remote collaboration. In *CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 315–322. ACM Press. 19, 47

Tang, J. C. and Minneman, S. L. (1991b). Videodraw: a video interface for collaborative drawing. *ACM Trans. Inf. Syst.*, 9(2):170–184. 46

Terrenghi, L., Kirk, D., Sellen, A., and Izadi, S. (2007). Affordances for manipulation of physical versus digital media on interactive surfaces. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1157–1166. ACM Press. 32

Thompson, J. (1973). *Beyond Words: nonverbal communication in the classroom.* Citation Press, New York. 71

Tse, E. and Greenberg, S. (2004). Rapidly prototyping single display groupware through the sdgtoolkit. In *AUIC '04: Proceedings of the fifth conference on Australasian user interface*, pages 101–110. Australian Computer Society, Inc. 28

Tse, E., Histon, J., Scott, S. D., and Greenberg, S. (2004). Avoiding interference: how people use spatial separation and partitioning in sdg workspaces. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 252–261. ACM Press. 156

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. In *AVI '06: Proceedings of the working conference on Advanced visual interfaces*, pages 336–343. ACM Press. 31

Ullmer, B. and Ishii, H. (1997). The metadesk: models and prototypes for tangible user interfaces. In *UIST '97: Proceedings of the 10th annual ACM symposium on User interface software and technology*, pages 223–232. ACM Press. 30

Veinott, E. S., Olson, J., Olson, G. M., and Fu, X. (1999). Video helps remote work: speakers who need to negotiate common ground benefit from seeing each other. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 302–309. ACM Press. 43

Vertegaal, R., Weevers, I., Sohn, C., and Cheung, C. (2003). Gaze-2: conveying eye contact in group video conferencing using eye-controlled camera direction. In *Proceedings of the conference on Human factors in computing systems*, pages 521–528. ACM Press. 44

von Hardenberg, C. and Bérard, F. (2001). Bare-hand human-computer interaction. In *PUI '01: Proceedings of the 2001 workshop on Perceptive user interfaces*, pages 1–8. ACM Press. 101

Wallace, G., Anshus, O. J., Bi, P., Chen, H., Chen, Y., Clark, D., Cook, P., Finkelstein, A., Funkhouser, T., Gupta, A., Hibbs, M., Li, K., Liu, Z., Samanta, R., Sukthankar, R., and Troyanskaya, O. (2005). Tools and applications for large-scale display walls. *IEEE Computer Graphics and Applications*, 25(4):24–33. 76, 92

Weiser, M. (1999). The computer for the 21st century. *SIGMOBILE Mob. Comput. Commun. Rev.*, 3(3):3–11. 20

Wellner, P. (1993a). Interacting with paper on the digitaldesk. *Commun. ACM*, 36(7):87–96. 20, 29, 47, 137

Wellner, P. (1993b). *Interacting with paper on the DigitalDesk.* PhD thesis, University of Cambridge. 29, 47, 137

Wellner, P. and Freeman, S. (1993). The doubledigitaldesk: Shared editing of paper documents. Technical Report EPC-93-108, Xerox Research Centre, Cambridge. 46, 47, 137

Whittaker, S. (1995). Rethinking video as a technology for interpersonal communications: Theory and design implications. *International Journal of Human-Computer Studies*, 42(5):501–529. 43

Williams, L., Kessler, R. R., Cunningham, W., and Jeffries, R. (2000). Strengthening the case for pair programming. *IEEE Softw.*, 17(4):19–25. 138

Wilson, A. and Robbins, D. C. (2006). Playtogether: Playing games across multiple interactive tabletops. In *Presented at IUI'06 Workshop on Tangible Play: Research and Design for Tangible and Tabletop Games*. 46, 47

Wilson, A. D. (2005). Playanywhere: a compact interactive tabletop projection-vision system. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 83–92. ACM Press. 42, 101

Wiseman, N. E., Lemke, H. U., and Hiles, J. O. (1969). Pixie: A new approach to graphical man-machine communication. In *Proceedings of 1969 CAD Conference Southampton, IEEE Conference Publication 51*, page 463. 63

Witmer, B. G. and Singer, M. J. (1998). Measuring presence in virtual environments: A presence questionnaire. *Presence*, 7:225–240. 166

Wu, M. and Balakrishnan, R. (2003). Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. In *UIST '03: Proceedings of the 16th annual ACM symposium on User interface software and technology*, pages 193–202. ACM Press. 33, 34

Yang, R., Gotz, D., Hensley, J., Towles, H., and Brown, M. S. (2001). Pixelflex: a reconfigurable multi-projector display system. In *VIS ’01: Proceedings of the conference on Visualization ’01*, pages 167–174. IEEE Computer Society. 92