Technical Report

UCAM-CL-TR-585
ISSN 1476-2986

Number 585

Personal projected displays

Mark S. D. Ashdown

March 2004

This technical report is based on a dissertation submitted September 2003 by the author for the degree of Doctor of Philosophy to the University of Cambridge, Churchill College.

ISSN 1476-2986

Abstract

Since the inception of the personal computer, the interface presented to users has been defined by the monitor screen, keyboard, and mouse, and by the framework of the desktop metaphor. It is very different from a physical desktop which has a large horizontal surface, allows paper documents to be arranged, browsed, and annotated, and is controlled via continuous movements with both hands. The desktop metaphor will not scale to such a large display; the continuing profusion of paper, which is used as much as ever, attests to its unsurpassed affordances as a medium for manipulating documents; and despite its proven manual and cognitive benefits, two-handed input is still not used in computer interfaces.

I present a system called the Escritoire that uses a novel configuration of overlapping projectors to create a large desk display that fills the area of a conventional desk and also has a high resolution region in front of the user for precise work. The projectors need not be positioned exactly—the projected imagery is warped using standard 3D video hardware to compensate for rough projector positioning and oblique projection. Calibration involves computing planar homographies between the 2D co-ordinate spaces of the warped textures, projector framebuffers, desk, and input devices. The video hardware can easily perform the necessary warping and achieves 30 frames per second for the dual-projector display. Oblique projection has proved to be a solution to the problem of occlusion common to front-projection systems. The combination of an electromagnetic digitizer and an ultrasonic pen allows simultaneous input with two hands. The pen for the non-dominant hand is simpler and coarser than that for the dominant hand, reflecting the differing roles of the hands in bimanual manipulation. I give a new algorithm for calibrating a pen, that uses piecewise linear interpolation between control points. I also give an algorithm to calibrate a wall display at distance using a device whose position and orientation are tracked in three dimensions.

The Escritoire software is divided into a client that exploits the video hardware and handles the input devices, and a server that processes events and stores all of the system state. Multiple clients can connect to a single server to support collaboration. Sheets of virtual paper on the Escritoire can be put in piles which can be browsed and reordered. As with physical paper this allows items to be arranged quickly and informally, avoiding the premature work required to add an item to a hierarchical file system. Another interface feature is pen traces, which allow remote users to gesture to each other. I report the results of tests with individuals and with pairs collaborating remotely. Collaborating participants found an audio channel and the shared desk surface much more useful than a video channel showing their faces.

The Escritoire is constructed from commodity components, and unlike multi-projector display walls its cost is feasible for an individual user and it fits into a normal office setting. It demonstrates a hardware configuration, calibration algorithm, graphics warping process, set of interface features, and distributed architecture that can make personal projected displays a reality.

Acknowledgements

I am grateful to Thales Research & Technology, part of the Thales Group, for their generous funding of my work and involvement in it. Chris Ash and Chris Firth oversaw the project, and Dave Orton constructed the second instance of the Escritoire hardware and helped to arrange the user tests.

I thank Peter Robinson, my supervisor at the University of Cambridge Computer Laboratory, and the other members of the Rainbow Research Group who have helped me with ideas and feedback.

Trademarks

Contents

List of Figures

Chapter 1

Introduction

In 1979 a decade of research at the Xerox Palo Alto Research Center culminated in the production of the 8010 computer, otherwise known as the Star [SIKV82, JRV$^+$89]. It was the prelude to the modern computing era: a personal computer with a mouse and keyboard for input, a graphical display for output, and an interface known as the *desktop metaphor* that used windows, icons, menus, and a pointer. Despite being a commercial failure, the Xerox Star defined the way in which computing resources were to be delivered to users for at least the next twenty five years.

Figure 1.1: Completing a tax return: paper still dominates many work practices.

The advent of the personal computer brought predictions of the paperless office, but this dream has not been forthcoming. Paper is still popular (Figure 1.1). One problem with the conventional interface it that it lacks space to display graphical information when compared to a physical workspace such as a desk. Another problem is that the desktop metaphor, which would more aptly be called the office analogy [Pem95], provided new interaction techniques inspired by the use of physical documents but did not supersede many of the existing ways in which those documents were used—people now use more paper than they ever did. A third problem is that the keyboard and mouse combination does not achieve the highest efficiency and ease of use for many tasks. Along with the absence of support for users who want to work together on a task, these points have motivated my work on display devices, input devices, and methods of interaction. These motivations are expounded below, followed by an outline of the rest of the dissertation.

1.1 Space

The space available on a real desktop permits a different style of work to that possible on the desktop metaphor. Figure 1.2 depicts an imaginary piece of furniture that I call the 9×12 inch desk, whose small size corresponds to the lack of space on a typical computer screen. Working with multiple documents on this desk would be annoying because there is only room for one sheet to be on top at a time. This is analogous to the memory hierarchy of a computer where the inability to store the working set of a program in the appropriate memory level results in *thrashing*. On the desk, documents must be brought to the top to be read and annotated, and thrashing occurs when lack of space forces them to be brought to the top consecutively rather than used concurrently. Visualization techniques can lessen the problem of lack of space—for instance, the use of multiple virtual screens provides a window onto a large virtual work space. However, I believe that ideally computer interfaces would provide the extent of physical space to which humans are accustomed in other forms of work.

Figure 1.2: The imaginary 9×12 inch desk. It is too small to work on, yet this lack of space is common in graphical user interfaces.

For decades computers have almost exclusively presented graphical interfaces on screens with diagonal sizes of 14 to 21 inches. Weiser [Wei91] envisaged ubiquitous computing accessed through tabs, pads, and boards—display devices of different sizes, posing varied challenges in implementation and interface design. The archetypal personal computer has a pad-sized display. There has recently been much interest in the tab-sized, small-screen technology of the many mobile devices being developed to exploit the economical and near ubiquitous access now available to digital networks [THSK01]. This type of interface is appropriate for nomadic users, but at home or in the office the size and versatility of a board-sized display can be exploited. User studies have shown that increasing the screen size used to display a conventional computer interface results in significant productivity benefits [CSR$^+$03].

Sight is the sense that supplies the highest capacity input channel to a human being. Ferguson [Fer92] recounts the history and successes of visual thinking, and rues the loss of traditional elements of engineering design (Figure 1.3) that has occurred as computer workstations have replaced traditional drafting tables. Information visualization, which exploits users' rapid visual processing capacity, has become an important part of computer science in recent years [Che02]. Screen space, although expensive, is worth investing in, as is evidenced by the amount of it installed in financial trading rooms (Figure 1.4) and increasingly in many other types of offices.

George A. Miller noted that the location of information in the world is important [Mil68]. We use, and are used to, spatial arrangement of information in the real world of offices, desks, papers, and books. The fact that space is unimportant to modern information processing systems should not mean that spatial arrangement is ignored, but rather that

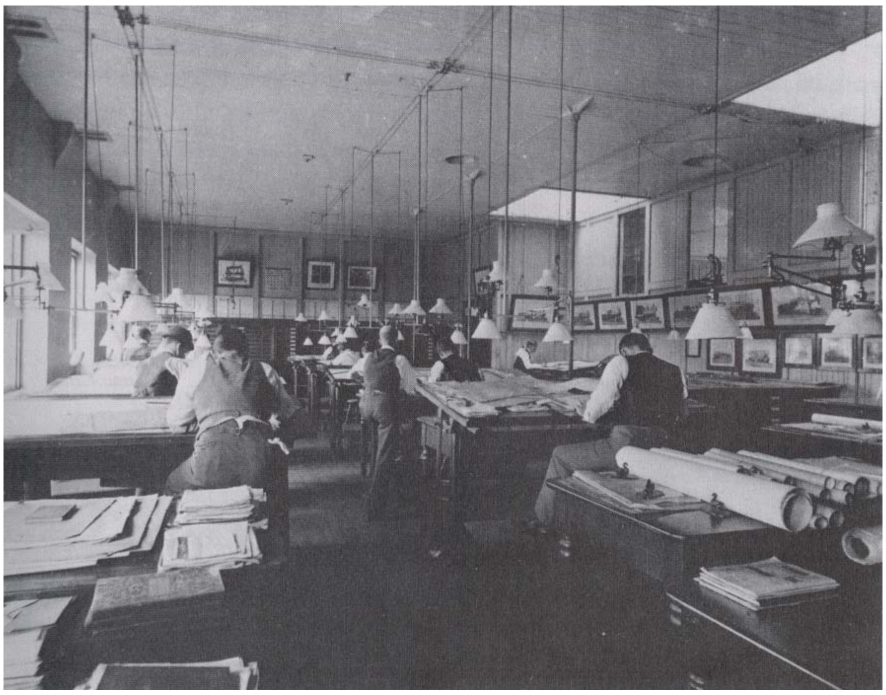


Figure 1.3: An example of the amount of space available in traditional work spaces. Large drafting tables are surrounded by documents and blueprints. Much space is available for viewing and recording information. The photograph, from Ferguson [Fer92], is of the drafting room of Baltimore & Ohio Railroad, 1899.

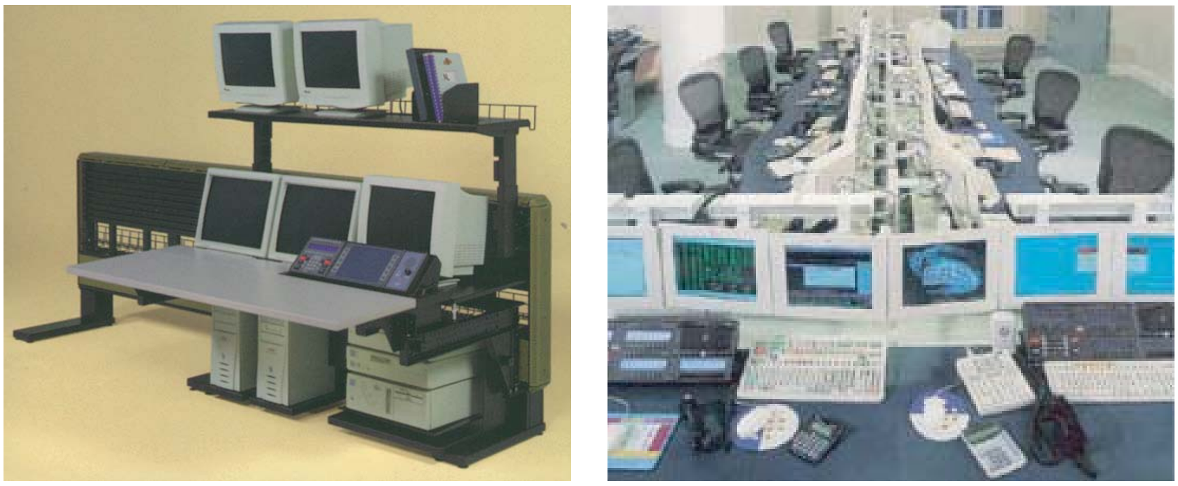


Figure 1.4: Screen space is a valuable asset. These are examples of the extensive screen space installed in financial trading rooms, where users must be continuously presented with the latest information. Photographs from `josephbrax.com` and `rinek.com`.

it is placed wholly at the disposition of the user. One of the reasons paper documents are still attractive compared to their digital counterparts is that they naturally support various types of spatial arrangement. Some other reasons are described below.

1.2 Affordances of paper

The *affordances* of an object are the perceived and actual properties that determine how it is used. The paperless office has not arrived because paper has affordances that have not been surpassed. Documents are still converted from electronic form to hard copy, albeit later in their lifetimes in many cases, and although highly structured tasks like searching a library are now accomplished through a computer rather than with a cabinet of cards, less structured work such as writing an academic paper that draws on ideas from various sources, generally still uses paper documents [SH97]. Similarly, a mechanical design can be simulated in the computer, but is more easily sketched on paper [AD01]. The use of paper in the modern world persists *because* of its unique properties, not in spite of them [JJK$^+$93].

The hardware of electronic books—batteries, memory, LCD displays, wireless networking, standard data formats, and user-interfaces—has progressed to the point where they seem to be a viable alternative to paper books, but their adoption now depends on other factors—distribution, encryption, availability of compelling content, copyright protection, billing, and collection of royalties [Har00]. Similarly, standards such as Adobe's Portable Document Format allow documents to be exchanged using an established representation, but before electronic documents can supplant paper the affordances of paper must be addressed. When virtual paper can do what physical paper can, its further capabilities, such as the possibility of sharing a document between geographically distributed, collaborating users, will become very useful.

Papers on a desk are cognitive artifacts—objects that expand the user's abilities by reminding them of pending tasks and providing easily accessible information [Nor93]. They are an example of knowledge in the world which complements knowledge in the head. As Donald Norman notes, the absence of such cues evokes the folk saying 'out of sight, out of mind' [Nor88, page 72]. Researchers at the Georgia Institute of Technology have been developing systems that automatically generate and display a montage of images showing a person's recent activities, to provide knowledge in the world to help her recover from interruptions to her current task [TM03]. Peripheral vision and kinæsthetic sense are important in early design work [Car93a]—a designer will place materials around a central work area so they can be glanced at quickly, or retrieved using the memory of the position of the hands that is necessary to grab them. In a recent study kinæsthetic cues were shown to aid spatial memory: users who positioned icons using a touch screen could better remember their positions than users who did the same task using a conventional mouse and monitor [TPSP02].

A study at AT&T Labs [WH01] found that despite the greater availability of digital information, workers kept large, highly-valued paper archives. It also found that only 49% of the documents were unique, the remainder being copies of publicly available data and unread information. Availability of information is important and people were reluctant to store paper in archives or rely on online sources because they would probably end up never using the documents again. Workers can be categorized as *filers* or *pilers* depending on their strategies for storing new documents. Contrary to the researchers predictions, filers amassed more information and accessed it less frequently then pilers. They argue that filers may engage in premature filing to clear their workspace, and archive information that later turns out to be of low value.

A study of workers at the International Monetary Fund found that when collaborating to create a document, which is a common task, they generally used only paper [SH97]. Paper was easily annotated by multiple people, it provided a large display area when the pages were laid out, and passing a document from one person to another was more satisfactory because a physical object changes hands and the two people meet and discuss their expectations of each other. One of the aims of Satchel [LEF$^+$00] was to allow people to pass documents to each other via *beaming*—impromptu transfer of electronic documents between people via mobile devices by transmitting encrypted tokens over an infra-red connection. Other affordances of paper have been addressed by simulating dog-eared pages [HS00] and allowing windows to be rotated through arbitrary angles [BL01]. The DigitalDesk [Wel94] and LivePaper [RR01] projects augment real paper with projected graphics. The use of horizontal displays like these affects working patterns, because they have different affordances to the conventional vertical screens of monitors and display walls [Kru02].

1.3 Input techniques

A new computer interface that is designed with regard to the affordances of paper requires new interaction techniques, especially when a large display device is used. A computer mouse has various features that make it a good input device for a pad-sized display: it traverses a continuous trajectory between any two points on the display; it provides a linear mapping between a user's hand movement and the pointer movement; and the entire width of an XGA (1024×768) display can be covered, with reasonable accuracy and control, by a single hand movement. However, when a display exceeds a certain size it becomes qualitatively different [SS97] and direct interaction with the display using a device like a pen is more desirable. The desktop metaphor does not work well on a board-sized display [Rek00]: menus are difficult to use when they are a long way from the user's centre of attention, and text displays are cumbersome when head movement is necessary to read the whole display. Control and feedback should be centred on the area to which the user is attending, as is the case with marking menus [KB94] and toolglasses [BSP$^+$93].

Although people do use two hands to type on a keyboard or press keys while using the mouse, the conventional interface does not allow interleaving or overlapping of continuous inputs with both hands. Bimanual input—using two hands—has manual benefits from increased time-motion efficiency due to twice as many degrees of freedom being simultaneously available to the user, and also cognitive benefits which arise as a result of reducing the load of mentally composing and visualizing a task at an unnaturally low level imposed by traditional unimanual techniques [LZB98]. In comparing the use of paper and electronic documents in a summarization task, O'Hara and Sellen [OS97] found that navigation of the paper documents was fast and automatic, unlike the electronic case. Among the techniques used to manipulate paper they noted: the use of two hands to overlap navigation with other activities; anticipatory page turning; the ability to lay out paper in space; and reading and writing spaces that could be accessed concurrently and manipulated independently. The fixity of information with respect to the physical pages was also significant, allowing the participants to acquire incidental knowledge of the location of information by reference to its physical location on the page, thus supporting searching and reviewing.

A bimanual interface should respect the difference between the dominant hand, which is usually the right hand, and the non-dominant hand, because they have different predispositions, and in most tasks the hands assume asymmetric roles. For example, in handwriting the non-dominant hand moves the paper into position, then the dominant hand writes portions of the text while the non-dominant hand intermittently adjusts the paper so the dominant hand does not have to move far from its resting position. Guiard [Gui87] defines the difference between the two hands with three principles: the non-dominant hand creates a frame of reference in which the dominant hand works, such as when one threads a needle; the movements of the non-dominant hand are coarser in space and time, that is, they are larger and less frequent; and the non-dominant hand acts before the dominant hand. These principles should be considered in the design of a bimanual interface.

Bimanual input requires the use of two pointing devices such as pens. Once the leap has been made from having a single pointing device to having two, there is little impediment to permitting an arbitrary number. MMM [BF91, Fre93] and co-operatively controlled objects [BBT97b, BBT97a, Bri98] show how fine-grained control between multiple users can be implemented. Once users can interact in this way, geographically separated users can be connected to allow distributed collaboration.

1.4 Collaboration

	same time	different time
same place	face-to-face (presentation support)	asynchronous (physical notice board)
different place	synchronous distributed (videophone)	asynchronous distributed (email)

Figure 1.5: Space and time taxonomy for computer-supported co-operative work, with example applications [EGR91]. Participants may be in the same place or different places, and may interact synchronously or asynchronously with each other.

The discipline of Computer-Supported Co-operative Work (CSCW) arose in the late 1980s when the near ubiquity of networked computers in organizations prompted researchers and developers to start thinking about systems that let people work together [GHR95]. CSCW Applications can be arranged into four groups [EGR91] depending on whether the participants are in the same place or different places, and whether they interact in real-time or through a series of disconnected events (Figure 1.5). Email, a system for asynchronous distributed communication, has been immensely successful and very widely adopted. The increasing power of personal computers, and growing capacity of long-distance network links between them, will allow systems like email to be joined by ones that support real-time interaction.

It is tempting to think that the goal of a system for synchronous remote collaboration should be purely to imitate a face-to-face conversation, but there are more effective ways to support many types of collaborative task, which may also exploit more effectively the strengths of the electronic medium [HS92]. The Picturephone from Bell Labs was introduced publicly at the 1964 World Fair and at the time it was predicted to replace the existing voice-only telephone by the early 1970s. The intuitive appeal of video communication fuelled positive forecasts of its wide-scale adoption [Egi88], but, except for the limited use of videoconferencing in business settings, it has not become a substitute for face-to-face meetings. This type of synchronous communication between participants via transmission of images of their faces provides what Buxton calls *person space*, but communication links can also be used to transmit video or other information to provide *task space* whereby the participants are copresent in a new virtual place—the domain of the task being undertaken (Figure 1.6). Kreuger called these types of interaction telecommunication and artificial reality [Kru83]. The difference between imitating a conversation and placing people in a virtual space is also pertinent to the design of collaborative virtual environments [BGR+98], which are generally used for entertainment.

Including a shared task space for users that are working together is important [Bux92] and studies have shown that for tasks other than negotiation a task space is more useful than a person space [ASM+00]. Fussel *et al.* performed studies where a novice attempts a task while being aided by a remote expert who shares a view of the work area [FKS00, KGF02]. They found the shared view helped by allowing the expert to monitor the novice's progress, by facilitating gestures between the participants, and by providing common ground for conversation which reduces ambiguity and enables utterances to be more efficient—deictic references are useful when a participant does not have a nomenclature for the objects in the task domain. They also found that task performance was harmed by a three-second latency in the transmission. They concluded that a shared visual space is essential for complex collaborative visual problem solving.

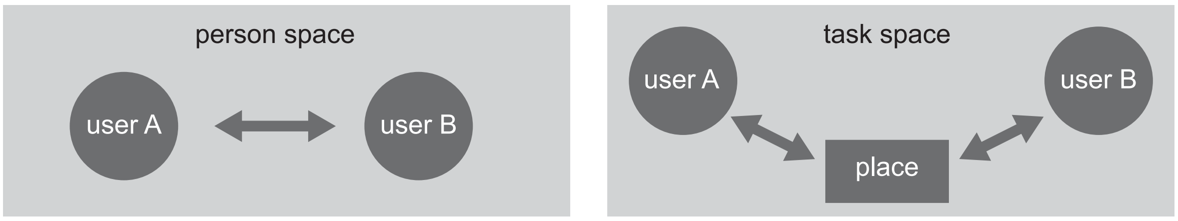


Figure 1.6: Person space versus task space: (left) a person space is provided by a video link directly between two users; (right) but a task space is a new domain in which the users can collaborate.

1.5 Objectives

I undertook the work described in this dissertation to address the points given above—lack of space in conventional interfaces, supremacy of paper in many tasks due to its unsurpassed affordances, manual and cognitive benefits of alternative interaction techniques such as bimanual input, and the need for collaboration that provides a shared task space for distributed users. I decided to build a large horizontal display that provides affordances and a user experience that are more like those of a real desk than those of the conventional desktop metaphor on a small vertical screen.

The new system has various requirements. It is intended as a replacement for the conventional workstation interface with its monitor, keyboard and mouse. The cost of the hardware should be feasible for a personal computer, so only readily available components should be used. The display should be created using projectors because they are standard pieces of hardware, and they can create both displays that are as large as a desk and displays that have resolution high enough to make an A4 page from a document legible. The interface should be controlled like a real desk using direct movements of the hands, rather than using the indirect pointing method of a mouse. Two-handed input should extend over the entire area of the desk. It should be possible

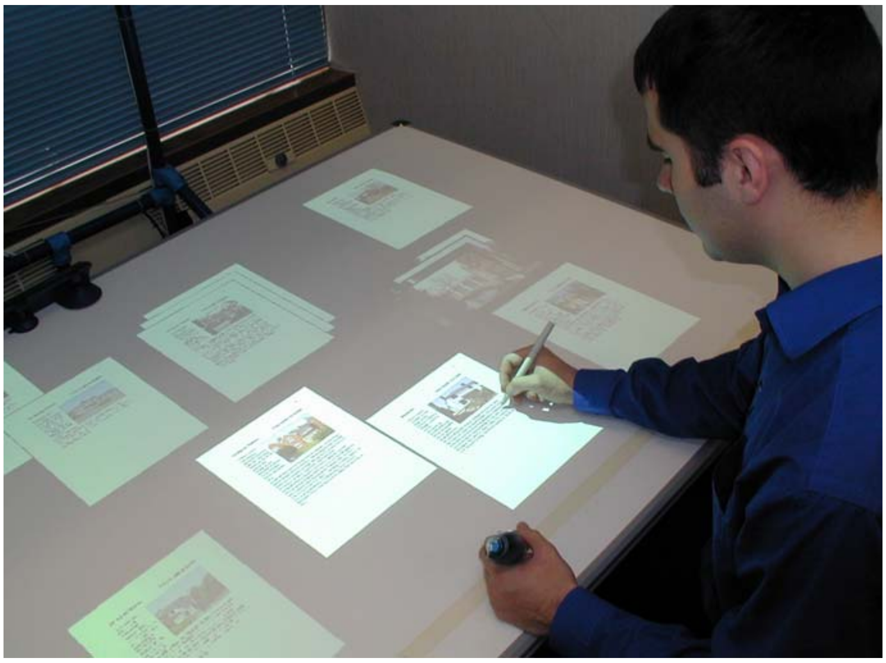


Figure 1.7: The Escritoire. It has a horizontal display the size of a desk that accepts input from a pen in each of the user's hands. One projector fills the desk, and a second creates a high-resolution area in front of the user for detailed work. Several desks can be linked via a network to allow collaboration in which the sheets of virtual paper can be manipulated by all participants.

to place existing data, such as documents and images, on the desk, and to read, arrange, and annotate them like sheets of physical paper. Multiple users should be able to use separate instances of the hardware to share the items on a desk surface and collaborate via the Internet.

By creating a new system with the requirements above I have addressed various questions about the design and implementation of such an interface for manipulating documents and collaborating using them. How can a projector system be arranged in a normal office setting to create a horizontal display that is as large as a desk and also provides sufficient resolution for standard documents to be read and annotated? The hardware should consist of standard components that are easily assembled and should not require precise mechanical adjustment—can a quick and easy calibration system be devised that allows software to compensate for inaccurate projector positioning? How can video hardware best be exploited to generate the graphics? What are the user interface issues with two-handed input over a large desk display? Can users usefully employ the large area? Can useful interaction be conducted in a task space formed by desks linked over standard Internet connection of the type available to most UK homes? The rest of this dissertation describes the theoretical and practical aspects of the hardware and software system I have created to address these issues, and the lessons I have learned while building it and performing user tests.

1.6 Dissertation outline

I have developed a system called the *Escritoire* (Figure 1.7) that uses the overlapping displays from multiple digital projectors to make a horizontal display that is as large as a traditional desk, but which still has high resolution in the area where it is needed, close to the user. Unlike other projector combinations which are used for presentations or high-definition scientific visualizations, it is a personal projected display for a single user. Thanks to recent reductions in the prices of digital projectors and increases in power of commodity video cards, and to the fact that the system can be driven from a single desktop computer, it is financially and practically feasible as an alternative to the conventional computer interface. The items on the desk are sheets of *virtual paper* that I have designed with the affordances of real paper in mind, so for example printing a document to sheets of virtual paper is just like printing a hard copy, and the document appears life-size on the desk. I have provided bimanual input by combining two pen input devices so they can be used on the same display, and have implemented and tested various interaction techniques. The system uses a client-server architecture that allows multiple users to collaborate on the same desk contents. I have augmented a standard videoconferencing system, which provides only a person space through audio and video channels, by allowing distributed interaction via multiple networked Escritoire desks. The new task space is shared by the users and through it they can collaborate on tasks such as reading, analysing, and annotating documents.

Chapter 2 lists projects and ideas that I have built upon when creating the Escritoire or that confront the associated challenges in different ways. Chapter 3 describes the hardware components and configuration of the Escritoire. The graphics supplied to the projectors that create the desk display are warped to compensate for the projector positioning and to avoid the need for mechanical calibration: Chapter 4 explains the mathematics involved, the calibration procedure, and the procedure by which the warping is achieved using commodity video hardware. Chapter 5 describes the mathematics and algorithms I have used to calibrate the various pen input devices. Chapter 6 shows how the Escritoire is split into a client that handles the input and output devices, and a server that stores, and enables interaction with, the sheets of virtual paper. It also gives the definition of the message protocol that is used to relay data between client and server. Chapter 7 describes two specific additions to the user interface—the ability to place sheets in piles, and pen traces that allow gesturing between remote participants—and also describes the user tests I have performed with the system and the subsequent results and insights. Finally, Chapter 8 presents my conclusions on all of this work and some ideas for future directions.

Chapter 2

Related Developments

The issues I have addressed to meet the objectives stated in Section 1.5 range from choice of hardware devices to human-computer interaction techniques, so I have called upon a variety of previous work. Visualization techniques allow large amounts of information to be displayed when extra screen hardware is not available—they make the most of a small display. In discussing future directions (Section 8.1) I explain how such a technique could be used on the Escritoire to relax the strict simulation of a real desk. Various large display devices have been built, and of particular interest are ones that exploit the user's visual periphery which is neglected by the conventional monitor. There are also several interesting human factors issues that will affect the design of a large display device. Projectors are currently the only way to make very large displays. They have been combined to make multi-projector display walls, and have also been used for augmented reality. Calibration of multiple projectors to planar surfaces and other objects is an important topic, and research on calibration techniques is still active. The conspicuous failure of the paperless office has led to two directions of research: augmenting real paper with computational properties, and simulating the properties of paper in an electronic document. I have taken the second direction which I believe will be much more powerful in the long term. Tangible user interfaces also support interaction that is more like the use of physical media than the conventional interface, and other input features such as pen input and two-handed input can offer further improvements. Finally, collaboration between users is an important feature that is available with physical media. Various groupware systems have been created that support real-time task-centred collaboration between users that are in the same place and users that are in different places linked by a computer network.

2.1 Visualization techniques

Very large displays have not been financially or practically feasible for personal workstations until recently, so various projects have sought to make better use of a conventional monitor. To make more information available on a small screen one can spread it out in time, or distort it in space, which leads to two types of visualization techniques exemplified by multiple virtual screens and focus plus context representations. One can also use methods that render graphical objects at different levels of detail, depending on how closely the user is attending to them, to make the best use of rendering computation.

2.1.1 Virtual screens

Rooms [JC86] was an early system that addressed the lack of space in conventional user interfaces. The disparity between the size and resolution of a physical desk or dining table, and a computer monitor, was used to motivate the creation of a window manager that presented multiple workspaces—rooms—that are created by the user to represent different tasks. Rooms had links between them represented by doors, so they formed a web that could be traversed. A window could appear in several rooms, and one room could be contained within another. Many window managers now support the concept of multiple virtual desktops.

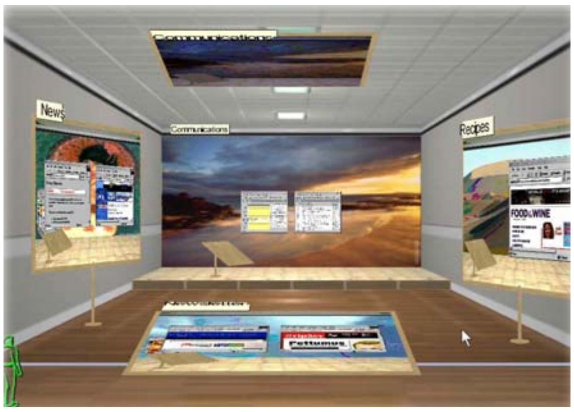


Figure 2.1: Microsoft's Task Gallery [RvDR+00] allows more tasks than usual to be managed on a conventional monitor by representing virtual workspaces as paintings in a 3D gallery metaphor.

Microsoft's Task Gallery [RvDR+00] takes task management a step further by presenting multiple virtual workspaces in a navigable spatial metaphor that is implemented with 3D video hardware and reinforced with animation. The walls of a gallery are adorned with tasks (Figure 2.1) like those in Rooms. Different background images remind the user of the differences between the tasks, and she can move backwards to view all of the tasks at once, or zoom in on a single task to view and control the application programs in it. Within one of the virtual workspaces, a *selected set* of windows is maintained, and the 3D hardware is exploited again to scale windows to avoid overlaps when a new window is added to the set. In the user's virtual left hand is a Data Mountain palette (described below in Section 2.4.2) that holds recently used documents and web links. Manufaktur [MG00] presents a similar interface in which multimedia documents are positioned in a 3D workspace where architects collaborate on a project.

2.1.2 Focus plus context

Focus plus context techniques [LA94] distort a graphical representation of a data set to create a detailed focal area while still accommodating the entire set on the screen. Fisheye views are an example of such a technique (Figure 2.2). They sacrifice the geometric relationships between graphical objects to display a particular group of them in more detail, and are often used for displaying vector diagrams such as graphs [SSTR93]. They are less suitable for bitmapped images because of the severe distortions. Various distortion functions can be chosen to set the relative size of the focus and context regions and the form of the transition between the two [Kea98]. A fisheye view would not be suitable for a system like the Escritoire because because it would greatly distort the desk contents which are stored as bitmaps, and it does not have the very predictable spatial layout that is necessary if items are to be grabbed instinctively from any part of the desk then brought to the high-detail region to be examined without distortion.

The Document Lens [RM93a] displays an entire multi-page document on a standard-size screen while simultaneously allowing the user to view any part of it at a legible resolution (Figure 2.3 (left)). The user moves a rectangular region in three dimensions to focus on a particular part of the document at a particular level of magnification.

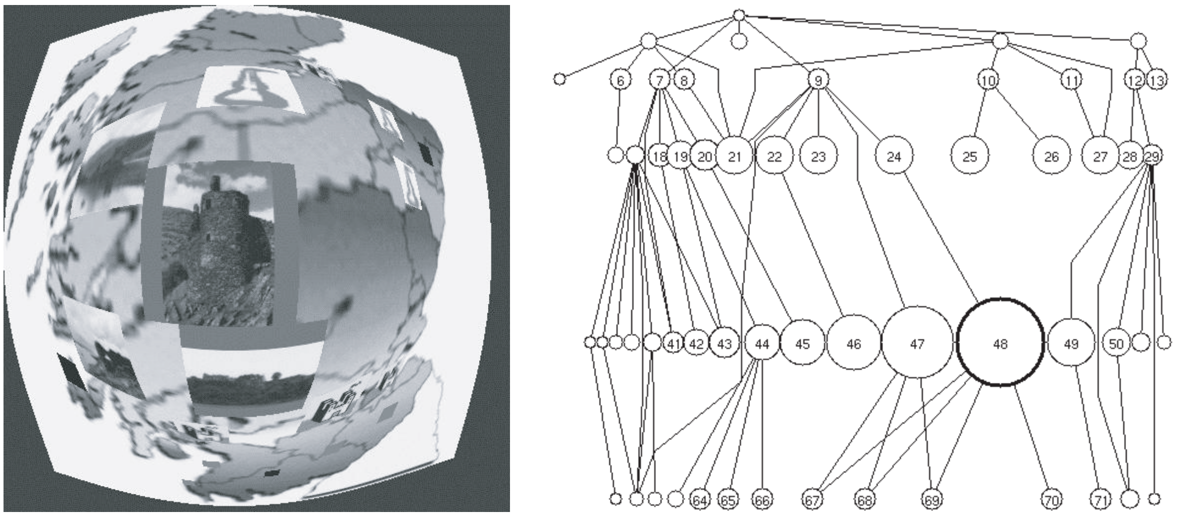


Figure 2.2: A fisheye view fits a large set of graphical information on the screen at the expense of the geometric relationships between its components: (left) an atlas with photographs representing the places [Kea98]; (right) a tree with node 48 in focus [SB94].

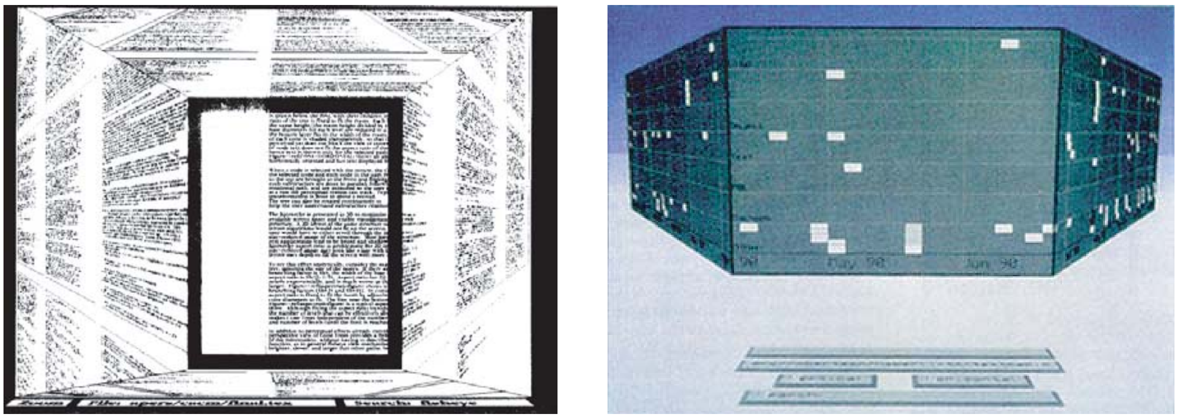


Figure 2.3: The Document Lens [RM93a] uses affine transformations to warp a document so the whole thing is visible on the screen while a movable focal area shows a portion in full detail. The Perspective Wall [MRC91] again allows a large amount of information to be viewed and maintains a smooth transition of items between the focus and context areas.

The parts of the document outside the high-resolution area are stretched efficiently using affine transformations to provide a continuous display of the global context. The Perspective Wall [MRC91] is a similar technique for displaying a wide visualization of information such as a time-line (Figure 2.3 (right)). The smooth transition of items from the detail area to the context is easily accommodated by the human eye. Both of these techniques make the form of the distortion function obvious.

2.1.3 Levels of detail

Pad++ [BH94] and its successor Piccolo [Pic] are zoomable user interfaces that present an infinite 2D plane on which graphical items are positioned. Rather than distort the information to fit it all on the screen, they concentrate on making it easy to continuously move between an overview of the complete set of items and detailed view of just one. When jumping between detailed versions of two objects the view zooms out to encompass both of them, then zooms in on the second one, so that the user can perceive the change of context. One of the technical challenges of this type of interface is to create software that can render a plane containing thousands of high-resolution graphical items at interactive rates—bitmaps are down-sampled and items at large distances are represented by basic shapes or omitted completely.

Three-dimensional level-of-detail rendering, whereby objects further from the viewer are drawn using a simpler polygon mesh, have become popular. For head-mounted displays, rendering in the periphery can be degraded to save computation without affecting task performance [WWHW96] because the user cannot see in such high detail near the edges of the display. Attentive displays [BDDG03] direct display and computational resources to where they are needed most, which is determined either by using heuristics to choose which areas of a scene contain the most interesting features or by using an eye tracker. Video streams can be compressed so that detail is retained in the most important areas, 3D scenes can be rendered so that they have most detail in the area where the user is currently looking, or non-photorealistic effects can be applied that place emphasis on the most important items in the scene.

A tool called the Restricted Focus Viewer [BJM00] creates a display on a standard monitor that has a small region shown in high detail while the rest of the screen is blurred. The aim of the program is to restrict the view to a movable rectangular window, so that the user's focus of attention around a diagram can be tracked without the use of expensive and unreliable eye-tracking systems.

2.2 Large displays

A conventional monitor, even a large one, only covers about 10% of a user's visual field, and about 1% of what one can see with a quick glance around. A simple way to employ more of the visual field using standard hardware is to add more monitors. A survey at Microsoft [Gru01] found that people who used multiple monitors came to find them indispensable, that windows are almost always positioned so they are wholly within one screen rather than straddling a boundary, and that secondary screens are generally used for peripheral information that can be accessed with a glance. The conventional user interface has problems when it is applied to a system with multiple large monitors, but some techniques have been developed to help: instead of scrolling the mouse repeatedly to move a window between monitors, the window can be *bumped* [RCMS03] from one window to the next by pressing a key; and a *high-density cursor* [BCR03] fills in the space between the current and last positions of the mouse cursor so that the user does not lose it when moving it quickly across a large display. These are useful modifications to the conventional interface but I think that the long term solution to lack of space, rather than combining multiple disjoint surfaces, will be to use larger display devices.

Funkhouser and Li [FL00] state 'Continuous rapid improvements in CPU performance, storage density, and network bandwidth have provided sufficient bandwidth and computational resources to support high-resolution displays and natural human-computer interactions. Nowadays, the main bandwidth bottleneck in an interactive computer system occurs in the link between computer and human, not between computer components within the system. Also, large-format display devices, such as projectors and flat panels, are rapidly becoming commodity items. New display technologies will revolutionize the way we use computers, making us rethink the relationship between information technology and our society. Consider how wall-sized displays enable qualitatively different human-computer interactions than traditional desktop displays.'

Various large display devices have been created, and some significant ones are described below. Currently the only way to make a very large, high-resolution continuous displays is with projectors. Plasma screens do not have high enough resolution, and they get very hot making them unsuitable for a desk system like the Escritoire. An interesting display technology for the future is one being developed by Screen Technology [ScT] whereby LCD displays are tiled, and an array of optical elements magnifies each display panel so they abut without a gap. Initial versions are due to be available in 2004. A user cannot attend to all of the information on a large display at once, so some of the items will form the focus while others will be peripheral to the current task. The description of display devices below is therefore followed by a review of projects that specifically target the visual periphery as support for a focal area. This is the hardware-supported version of the focus plus context and level of detail techniques in Section 2.1 above. There then follows a description of the human factors issues specific to large displays.

2.2.1 Large display devices

Early attempts to build computer support for meeting rooms included whiteboard-sized displays to provide a shared focus of attention, but these were controlled at a distance using a keyboard and mouse. Then Xerox PARC produced the Liveboard (Figure 2.4), which has a cordless stylus and a large vertical back-projected display measuring 4×3 feet, and displays the X Window System via a special driver. The original device had 1120×730 pixels giving it a resolution of 24 dpi. Elrod *et al.* [EBG+92] found that the most popular programs on the Liveboard were the whiteboard and the slideshow with annotation, and that the improvement users would have liked most was higher resolution. Tivoli [PMMH93] is a whiteboard application for the Liveboard, designed to support meetings where participants work together on an idea, rather than for presentations. Core functionality was identified and implemented: simple pen and gesture scribbling and editing, multiple pages, saving and retrieving, printing, and importing images. Extended functionality was anticipated as future work: remote collaboration, meeting management tools, and integration with other ubiquitous computing devices. Tivoli prompted reconsideration of many tacit assumptions of traditional GUIs. A pen allows natural gestural input, does not require a cursor, and generates direct position data unlike the relative movements of a mouse that can be scaled to alter sensitivity. A large surface necessitates careful positioning of buttons, menus, and pop-up messages, and places emphasis on keeping user control at the pen through techniques like gestural commands. Work on Tivoli has continued [MCvM97] with emphasis on the structure

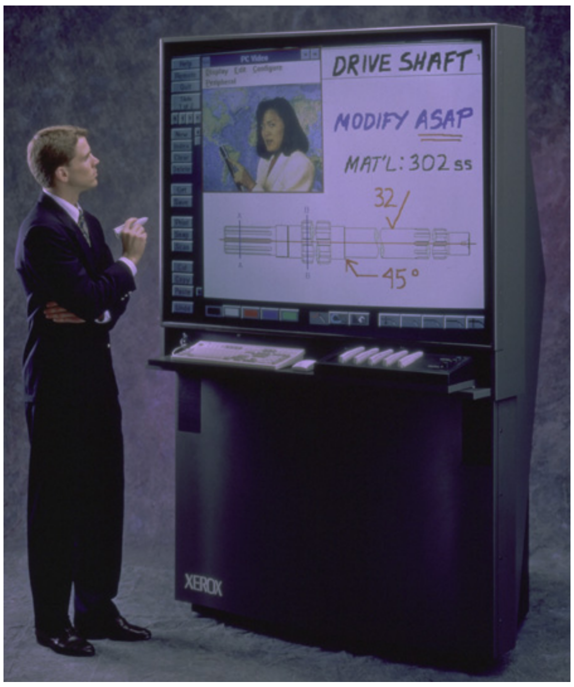


Figure 2.4: The Xerox Liveboard is an example of a large display device. The original was a 4×3 foot, 1120×780 pixel, rear-projected display driven by the X Window System and controlled by a laser pen tracked by a camera behind the screen [EBG+92].

and manipulation of object-oriented data to allow selection, scaling, and grouping of items; zooming as in Pad++; splitting of the board into regions using horizontal and vertical lines; and annotations that can be iconified. Tivoli has also been augmented to handle structured data, rather than just strokes stored as bitmaps [MvMC98, McM00].

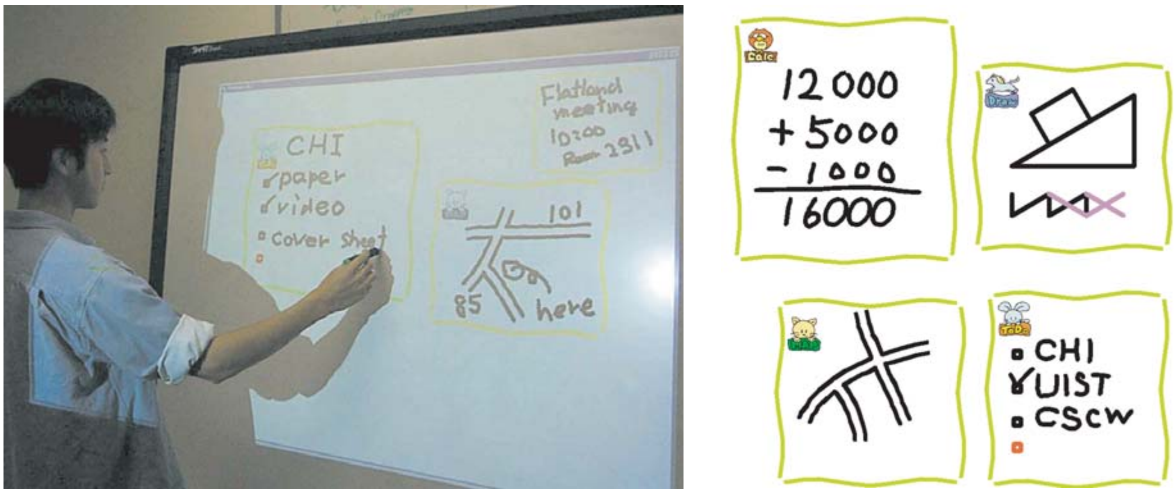


Figure 2.5: The Flatland system [MIEL00]: (left) the user writes on the projected display as if it is a whiteboard; (right) a different behaviour is applied to each segment so that the strokes are interpreted as, for example, a calculation, or a list.

Flatland [MIEL99, MIEL00] is formed from a digital projector and a wall-mounted board with a stylus. It behaves like a normal whiteboard, displaying information and allowing writing, but the surface can be split into segments by drawing straight lines, information can be added to a segment by drawing inside it, and segments can be given behaviours so strokes are interpreted, for instance, as parts of a map (Figure 2.5). As the active segment fills up with information it grows, and inactive ones shrink, so that more information can be fitted on the board without erasing anything.

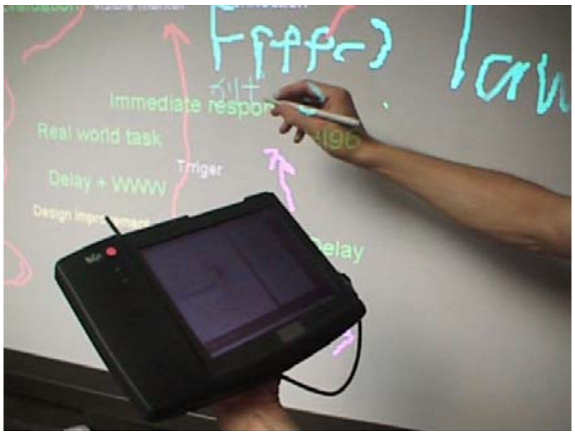


Figure 2.6: The pick-and-drop operation [Rek97] allows a PDA to act as a private workspace for a larger public display. During a presentation the speaker can drop prepared material onto the board as the need arises.

The creation of IdeaBoard [NOT$^+$96], a whiteboard for educational applications, lead to some guidelines for GUI design: the display should be operable from any position so the user does not have to move across the large display to get to an interface widget; very large and very small movements should be avoided because they are hard to do; direct pointing and manipulation are good because they navigate the audience's focus of attention; the display should change continuously so the audience can follow the changes. The creators describe several new widgets: a window that is dragged using a scaled position relative to the start of the drag so it can be moved quickly through a large distance, a scroll bar around the edge of a window, a toolbar that expands to fill the width of the window, a counter that is dragged up and down to choose a number, tabbed pages that are previewed by moving the pen over the tabs, and pages that are turned by dragging the pen across them.

Rekimoto has developed a multiple-device approach for supporting whiteboard interactions [Rek98] because the large surface makes traditional GUI design ineffective. A PDA is used as a private workspace on which detailed material can be prepared before it is brought to the board, so the user can make a presentation by successively moving previously prepared items from the PDA to the large display, or browse help files for a program without the audience seeing. This combination of board-sized and tab-sized devices is an example of Weiser's vision of ubiquitous computing [Wei91]. A technique called *pick-and-drop* (Figure 2.6) is used to transfer items between different devices using the same pen [Rek97]. The technique is also useful for transferring files between computers—transferring files between locations on a single computer is simple with a direct manipulation interface, but transferring between computers is more difficult because symbolic information like network identifiers for computers must be used and shared directories may have to be created. Pick-and-drop allows direct manipulation between physical devices. Another technique called *scoop-and-spread* [AMR00] uses gesture commands from a PDA with a tracking device. A set of images stored in a PDA can be spread over a projected horizontal display by swooping the PDA in a gesture across the surface, or *scooped* up with a corresponding gesture.

2.2.2 Visual periphery

Early work on exploiting the human visual periphery with computer displays was conducted in the MIT Media Room. Dataland [Bol84] presented the user with graphical items such as maps, letters, calendars, and photographs, on a monitor. The monitor display also had a movable *you-are-here* marker that acted like a magnifying glass: the portion highlighted by the marker was displayed in high detail on the rear-projected display that filled a wall of the Media Room. A related project, the World of Windows [Bol81], presented an ensemble of simultaneous video streams on the wall (Figure 2.7 (left)). Such a deluge of information would ordinarily be overwhelming, but the user's gaze direction was used to identify the stream on which they were focused, then the volumes of the other streams were reduced. The volume of the soundtracks could be

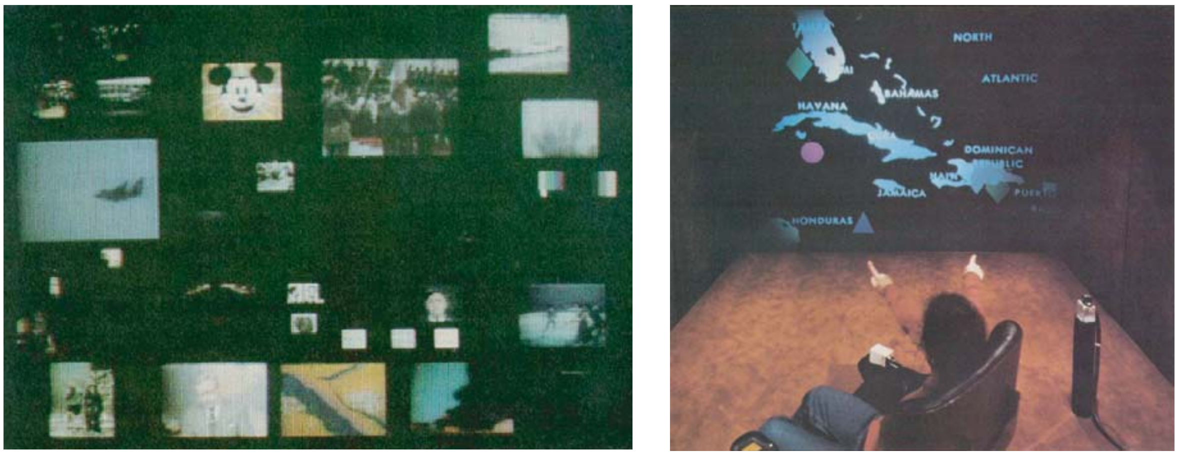


Figure 2.7: Early work on very large displays was done in the MIT Media Room: (left) the World of Windows [Bol81] presented the user with a wall full of video streams from which one could be selected to view in detail; and (right) Put-that-there [Bol80] used voice recognition and deictic gesturing to modify the items on the wall-sized display.

scaled by their distance on the screen from the focal video stream, or by their temporal distance from it so that less recently focused streams were quieter. Following a period of prolonged attention the focal video stream was enlarged to fill the whole display. Put-that-there [Bol80] made pronouns and deictic gestures a part of the user interface. A six degree-of-freedom tracker was attached to the user's finger, and speech recognition was used. The user could then point successively at an object on the large display and then at a new location, and say "put THAT, THERE" (Figure 2.7 (right)).

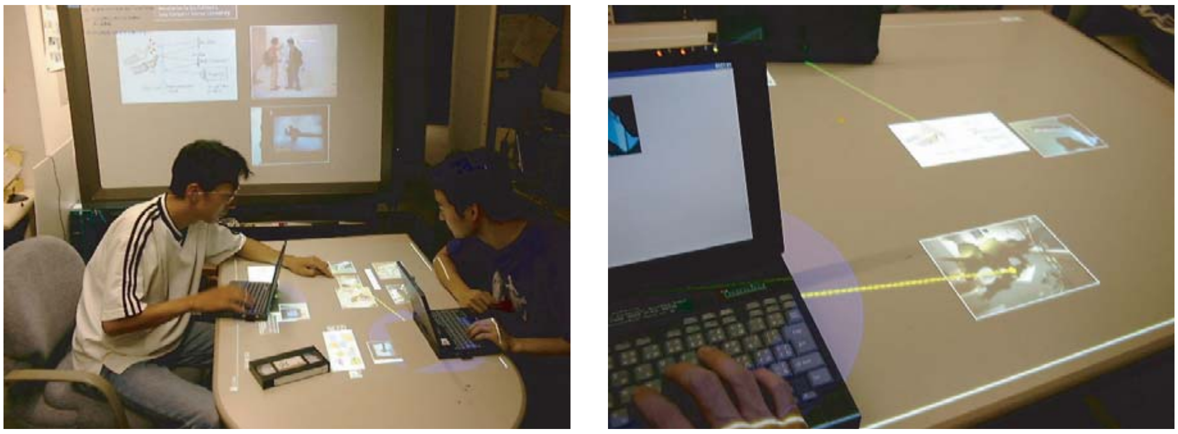


Figure 2.8: In Rekimoto's (left) augmented surfaces [RS99] items can be (right) hyper-dragged out of a notebook computer using the mouse, and placed onto a projected display on the table or wall.

Guimbretière [Gui02] has made an interface for sketching and arranging material on the Stanford Interactive Mural, a 6×3.5 foot wall display described below in Section 2.3.1. Despite the large size of the display he has used a visualization technique called Zoomscapes [GSW01] to fit on more items. Users find it natural to push items, and groups of items, that are not being used to the top of the board where they remain visible to the other people present, so the Zoomscape scales the items, based on the location of the pen as they are dragged, so that items in the top fifth of the screen shrink to one quarter of their original size. There is a ramp in the scale factor between the large-scale and small-scale areas. Users liked the fact that items can be left to overlap both regions if they are dragged accordingly.

The human eye sees in high resolution in a small area called the fovea in the centre of the visual field, and the resolution quickly drops off away from that centre. Also, objects that are close fill a larger angle at the eye than objects that are farther away, so are seen in higher detail. These points mean that the hardware of a large personal display for a stationary user will ideally be arranged to place most of the pixels in a focal area near to the user.

Several projects have created displays with multiple disjoint surfaces of different resolutions that have different sizes and distances from the user. Courtyard [THY$^+$94] used a large wall display to show information to a group of users, and allowed a user to drag a window down onto his private monitor to work on it. A system of augmented surfaces by Rekimoto [RS99] uses a notebook computer as a user's focus, while lower-resolution projected desk or wall displays create the surfaces that form the periphery. Cameras locate and register items on the surfaces using printed visual markers—2D glyphs similar to bar codes [RS99]. Projected graphics add information to the identified objects (Figure 2.8 (left)) and *Hyperdragging* (Figure 2.8 (right)) allows an object to be dragged off the edge of an identified notebook computer and onto the desk or wall. Using only the readily available knowledge about their physical locations, a user can transfer an item between two computers by hyperdragging it between them. Kimura [MMV$^+$01] adds a large projected wall display to the conventional monitor to show representations of the user's tasks, which reminds her of the background tasks that are not currently visible on the monitor. Information about a task is inferred using the Win32 Hooks API in Windows, and a graphical depiction allows the user to quickly identify each task displayed on the wall and switch to it. The system incorporates context awareness so, for instance, when a person enters your office the tasks that are associated with that person are displayed more prominently in anticipation of an informal meeting [VDMC02].

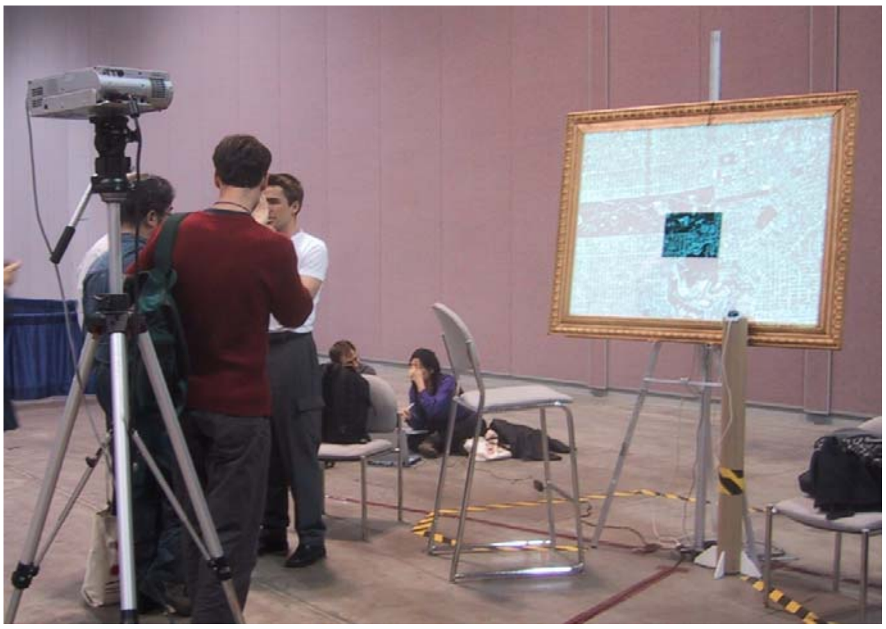


Figure 2.9: The focus plus context screen [BGS01] is a standard front-projected display with an LCD monitor set into the screen to create a high-resolution focal area.

Feiner and Shamash [FS91] combined a low-resolution transparent head mounted display and a magnetic tracker with a normal monitor to create a display with areas of different resolutions. It allowed windows to be placed in a peripheral region off the side of the monitor, using a modified X server which displayed the windows as simple rectangles with titles when they were outside the high-resolution region covered by the monitor. The researchers were motivated by the fact that text editors are never used in virtual reality because the resolution is not high enough, but with their system one can move windows to the monitor to view them in detail. Robinson alluded to combining projectors to make large continuous displays with high resolution areas where they are needed [Rob95, page 10], and recently Baudisch *et al.* have combined an LCD monitor and digital projector to make a *focus plus context screen* [BGS01, BG02] which has high resolution in the LCD area and much lower resolution in the surrounding projected region (Figure 2.9). They use a conventional window system with a very large framebuffer, then crop off the excess to get a bitmap for the focal region, and down-sample the entire image to get one for the peripheral region [BGBS02]. This system is a large multi-resolution display device like that of the Escritoire, but it runs conventional application programs which are controlled using the standard keyboard and mouse, and calibration is purely manual. The lack of graphics warping means that the projector must be precisely positioned, which restricts the location of the user.

2.2.3 Human factors

The size of a display affects how it is perceived and used. A study measured the effects of size and resolution of a display on sketching and sorting tasks [EH00, EH02] by comparing three displays which had the same number of pixels: a projected desk display, a tablet display with stylus, and a conventional monitor. Several participants found the desk to be too large and the tablet too small. The researchers suggest that interfaces designed for large work surfaces should put primary information near to the user and less important information in the periphery. Alignment of the pen on the desk display was a problem for some users because of parallax errors due to the thick glass of the back projected surface.

Large displays have been found to improve performance in spatial tasks [TGSP03] because they give an increased sense of presence. The choice of a large display over a conventionally-sized one can also have social effects [RDS02]. While using projected displays, Tan and Czerwinski [TC03] found that information physically situated away from the user or on walls is tacitly assumed to be public and visitors to the office have no hesitation in reading it. This may be an advantage: field studies of workers who collaborate on tasks that involve paper documents [LHG92] have indicated that one of the reasons paper persists in those situations is because it allows a person's co-workers to remain aware of the general form of his activities. This allows co-located workers to act as a team, even when they are performing seemingly individual tasks that do not depend directly on each other. When compared to a desk of papers, a large computer display also has the advantage that at the touch of a button any sensitive documents can be hidden, rather than left accessible when the owner leaves her office.

2.3 Projectors

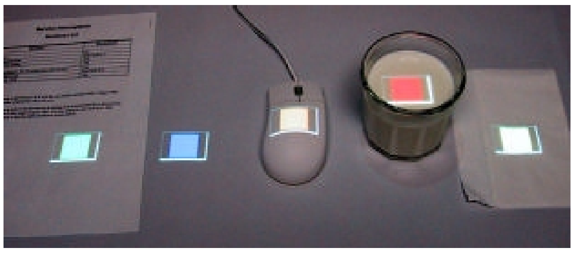


Figure 2.10: Users will react differently to displays projected onto different objects. In an experiment by Podlaseck *et al.* [PPA^{+}03] where coloured buttons were projected onto the items in the figure, a surprising number of users did not perceive the display projected onto the glass of milk because they did not expect it.

I have used projectors to make the display of the Escritoire because they offer high enough resolution for detailed work rather than just presentations, because small portable units are now available, and because they are becoming affordable as an addition to a personal workstation. Resolutions are not high enough to fill a desk while also rendering a life-sized PDF file at a legible quality so I have combined projectors to make a display with regions of varying size and resolution.

Various projects have combined projectors to create multi-projector display walls for immersive displays or visualization for groups of people. Another use of projectors is for augmenting physical objects with projected graphics. There are various geometric and photometric issues in calibrating projectors for these applications. Multi-projector displays, augmented reality, and calibration are described in the sections below.

Recently projectors have also been used for other applications, such as a very large display for a conventional desktop computer [BW00, Wel], and immersive teleconferencing for a tracked user [TCY+02, YKN+02]. Majumder and Welch suggest several novel ways to use two or more projectors that are pointing at the same screen and overlap completely: have one projector focussed and the other unfocussed, then either create depth of field effects by rendering the image twice and sending the results to the separate projectors, or make a blurred image with a focussed inset area; create parallel optical pipelines with their own projectors, then render and display, for instance, diffuse shading with one pipeline and specular with the other; or simply make a display with higher contrast and more intensity levels.

There are psychological issues to consider with projected displays, especially when the graphics are projected onto general objects rather than flat white screens. Podlaseck *et al.* [PPA+03] identified *functional fixedness* for projected displays, which is related to Norman's concept of affordances (Section 1.2). They projected coloured buttons onto various objects including a glass of milk (Figure 2.10) and asked subjects to point at the colour they liked the most. Around 20% of the subjects never chose a colour projected onto the milk, and in subsequent questioning 10% could not even remember seeing a glass of milk at all. Their preconceptions about the functions of objects influenced their perception.

2.3.1 Multi-projector displays

The PowerWall [PW] was an early multi-projector display wall. It measured 6×8 feet and had eight megapixels, giving it a resolution of 34 dpi. CAVE environments [CNSD93] are used for immersive virtual reality as an alternative to head mounted displays. CAVE stands for CAVE Automatic Virtual Environment. Graphics are rear-projected onto the walls of a room in which the user stands, and the user's head is tracked so that he can be presented with a view with the correct perspective. An important advantage of using the CAVE is that the user can see his own body in the virtual environment.

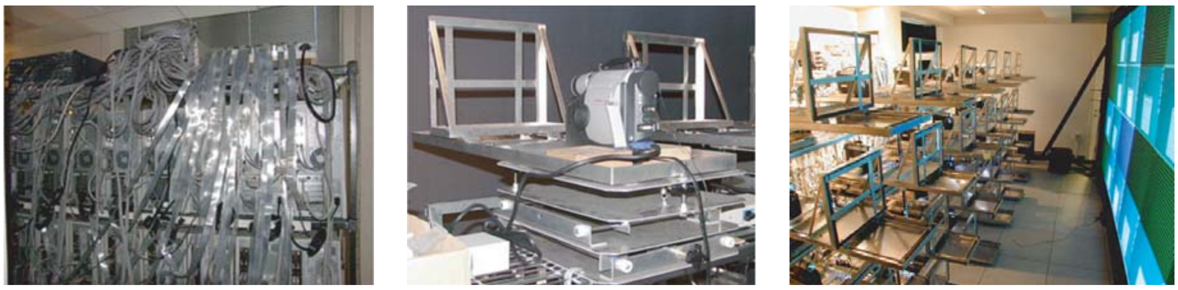


Figure 2.11: The Princeton display wall: (left) the cluster of PCs that drive the projectors, linked by a fast network; (centre) a projector on its mounting that has six degrees of freedom and a shadow mask to restrict the projected image to an adjustable rectangular area; (right) the array of 24 projectors.

Many groups have combined multiple digital projectors to form display walls for presentations or scientific visualization. The InfoWall [WSK+00] at AT&T Labs is an eight-projector display used to visualize large data sets from telephone, cable television, and Internet networks. The Interactive Mural at Stanford [HH99, GSW01] is a six-foot wall display delivering approximately 57 dpi that uses 12 projectors driven by a cluster of 32 Linux PCs connected via a 1Gb/s network. The Princeton display wall [LCC+00] has 24 projectors driven by a cluster of PCs running Windows NT (Figure 2.11 (left)), which are linked via Myrinet [My], a high-speed network technology. The projectors are mounted on custom-made adjustable bases (Figure 2.11 (centre)), and they combine to form a 18×8 foot 28 dpi display (Figure 2.11 (right)). A group at the University of North Carolina made PixelFlex [YGH+01], a reconfigurable front-projection display formed with eight projectors that have pan-tilt mirrors so that the projected graphics can be pointed at different locations on the screen. WireGL [HEB+01] and its successor Chromium [HEB+02] have been developed to send OpenGL [WND99] procedure calls to a set of networked computers to perform distributed rendering. They allow 3D applications to be displayed on multi-projector display walls.

2.3.2 Augmented reality

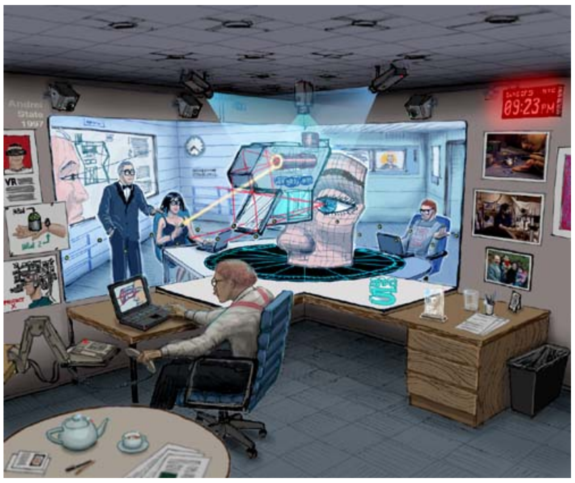


Figure 2.12: A sketch of the Office of the Future [RWC+98]. The researchers envisage a multitude of inexpensive cameras and projectors embedded in the office. The cameras infer the geometry and reflective properties of the visible surfaces and the projectors create 3D imagery on them for a user whose eye positions are tracked in 3D.

Researchers on the Office of the Future project [RWC+98] point out the falling cost and size of digital cameras and projectors, and predict a time when an office can be augmented with a multitude of such inexpensive devices to provide an immersive display for a tracked user (Figure 2.12). The aim of the project is to develop a unified set of computer vision and computer graphics techniques to create spatially immersive displays for a moving viewer from arbitrarily positioned, overlapping projectors shining onto irregular display surfaces. Calibration of the display involves geometric registration and intensity normalization [RWF98]. A camera observes the projected location of pixels from several projectors, and extracts depth and reflectance information. The extent of a projector is determined by projecting a white rectangle then thresholding the image from the camera, and intensity blending functions are used in the overlaps. High-resolution, wide field-of-view displays are envisioned that are easy to set up, use, and maintain, and a panoramic display has been made as a proof of concept [RBY+99]. The group has also made a system that creates life-sized physical scenes that the user can walk around inside, by projecting onto an approximate physical version of the virtual scene that is assembled from white blocks [LWLF01].

Augmented prototyping [VdSPvG03] involves projecting properties onto an object to experiment with design choices. Figure 2.13 shows a simple model of a car that is augmented with properties such as the paint colour. Raskar *et al.* have made a similar system for simulating dynamic motion from a static model of a car [RZW02]. They use non-photorealistic rendering and have implemented various effects, such as translating the background vertically and moving the shadows to make the car look like

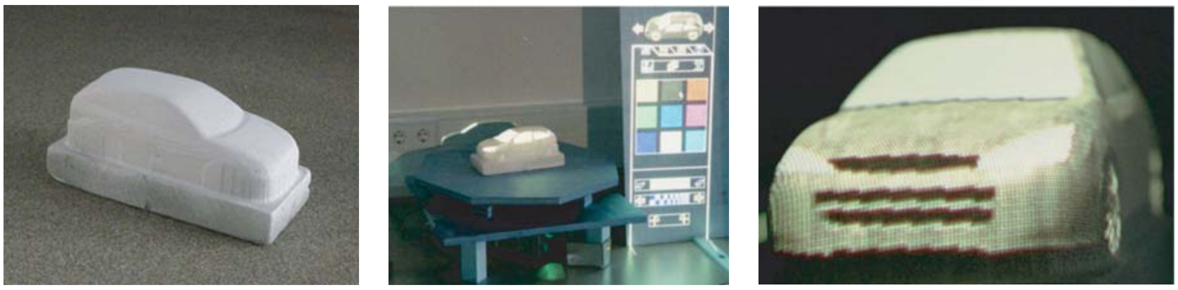


Figure 2.13: Projections can be made onto complex objects rather than flat surfaces. In augmented prototyping [VdSPvG03] a model of a car is augmented with projected graphics. The user chooses attributes, such as the paint colour, from a projected menu, and views the results on the model.

it is jumping into the air. Both systems use manual calibration of the virtual object to the physical model.

The Everywhere Display [Pin01a, Pin01b] combines a projector and a pan-tilt mirror to form a steerable display that can augment various surfaces in the environment with projected graphics. A camera in the corner of the room is used to track users' heads as they move around the room [PPL$^+$02, PKL$^+$02] so that displays can be made to appear near them and to avoid a user obscuring the display. The researchers have added the Everywhere Display to a simulated retail environment and are studying the interface issues—for instance, with the current version of the interface people did not realize the intended relationship between a projected display and the physical objects near it [SPK$^+$03a, SPK$^+$03b, SPK$^+$03c]. Sato *et al.* [TIS$^+$03] are working on a system to continuously scan an office environment with a laser range finder and a colour camera, which would allow a steerable display to react to changes in the environment by always choosing a projection surface that is available and not occluded by any objects in the room.

2.3.3 Calibration

Until recently multi-projector displays were calibrated manually. For instance, the Interactive Mural has masking tape to restrict the projector frustums, projectors are made orthogonal to the screen by projecting a series of lines onto a print-out of similar lines and viewing the moire pattern, and coloured rectangular edges on the projected images are used to gauge when the pixels of adjacent projectors are precisely aligned [Gui02].

Sukthankar *et al.* [SSM00, SSM01] have developed a method for efficient geometric calibration of an obliquely projected display, which obtains planar homographies between camera, projector, and screen, then warps the images before they are projected. This is closely related to the calibration method I have developed, which is described in Section 4.1. They have also extended the system to the automatic calibration of multi-projector display walls [CSWC01, CSWL02], and have used multiple projectors to eliminate shadows when a person walks in front of the screen [SCS01] and suppress the light that is falling on the person so they are not dazzled [CSRS01, CRSS03]. The shadow elimination and occluder light suppression currently run at approximately 3 Hz so they are not yet ready for deployment in a product. A single projector can also be calibrated to create a display that spans multiple planar surfaces [AS03, AFSR03]. Tracking of fidicules at the corners of the screen and sensing of the current position of the projected display

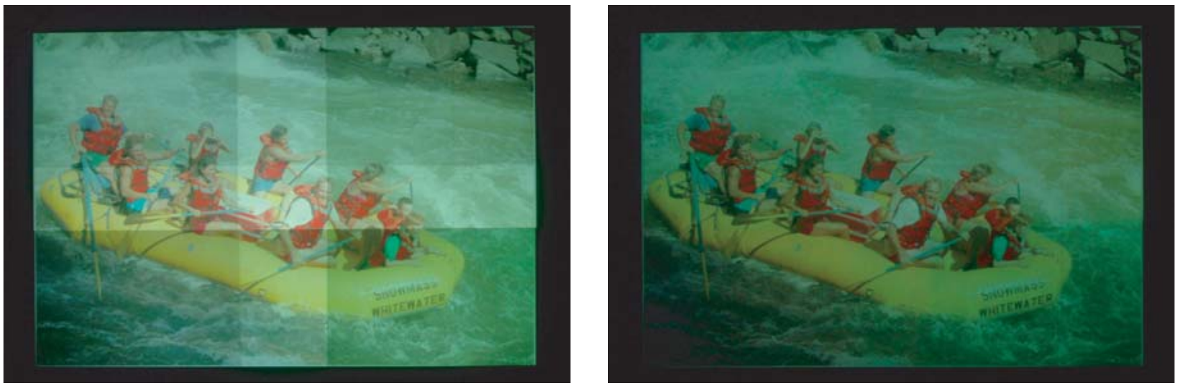


Figure 2.14: Photometric calibration of a multi-projector display [MS03]. Without calibration (left) the 2×2 projector array is much brighter in the overlap regions. The images from the projectors are selectively attenuated (right) to achieve uniform brightness across the whole display.

can allow a projector to be moved while keeping the projected image in the same place [RFC+02].

In addition to the geometric calibration issues, there are also photometric ones because luminance and chrominance must be matched between projectors. Intensity blending in the overlap regions between adjacent projectors is important for multi-projector displays [RZW02]. Projectors based on DLP (Digital Light Processing) technology from Texas Instruments contain a chip with tiny mirrors on it that reflects light from the lamp through coloured filters on a spinning wheel and onto the screen. Colour matching between such projectors is problematic because, unlike other types of projector that have only red, green, and blue components, they typically also have a white component because the manufacturers use a clear segment in the colour wheel to boost the intensity rating of the projector at the expense of colour fidelity.

One option is to use look-up tables to adjust the RGB components of the image, as in gamma correction, then use a 3×3 matrix to simulate the way in which the components interact [Sto01, Sto]. Majumder and Stevens [MS03] have developed a photometric calibration system for multi-projector display walls that uses a single camera to measure the luminance and chrominance properties within each projector's display region and the differences between the projectors—the intra and inter projector variations. They use an intensity range for the whole multi-projector display that can be achieved at all points on the display (Figure 2.14). They are currently working on reducing the extent to which limiting the intensity range in this way wastes resources. They apply a Luminance Attenuation Map to each projected image to account for the difference in luminance across the ouput of each projector, and use Look Up Tables (LUTs) to linearize the response of the projectors to the three colour channels. The method currently takes 2 to 3 seconds per frame but that will be greatly improved when modern video hardware is harnessed to implement the LUTs in hardware.

Wellner envisaged a device called the DigitalDeskLamp that could be pointed at surfaces in the office to reveal digital information [Wel94, pages 80–81]. To get higher resolution the user would simply move the device closer to the surface. Raskar *et al.* [RvBB+03] have recently created such a device, which they call an iLamp, by putting a projector, camera, tilt sensor, computer, keyboard, and wireless network adapter in one box. It identifies glyphs on objects using the camera, then augments the objects with projected graphics. Calibration is aided by the tilt sensor that resolves any ambiguity about which way is up.

2.4 Natural interfaces

Many people have attempted to create a computer interface that is more natural and intuitive than the conventional desktop metaphor. Pen and paper form a universal and natural medium for disseminating and recording information—reading and writing are easy—so attempts have been made to support the affordances of paper for access to digital information, by augmenting paper with projected graphics, and by simulating on a conventional computer some of the details of working with paper. Researchers have also built tangible user interfaces whose mechanical manipulation has much more variety than that of the keyboard and mouse, and they have developed alternative input devices and techniques like bimanual input.

2.4.1 Augmenting paper

The DigitalDesk [NW92, Wel93a, Wel94] combined real paper and projected digital information on a horizontal desk surface. The user could copy and paste information from paper documents, via an overhead camera and Optical Character Recognition software, to electronic documents or programs (Figure 2.15). One could also copy graphics from a physical book of clip art and include them in an augmented picture that one was drawing on paper. Further work investigated animated paper documents [RSW$^+$97, Rob99] where the pages of a paper book are recognized with computer vision, then augmented with extra properties. For instance, an applet could be added to a page about graphs to rate a user's performance at sketching a particular mathematical function.

Later systems pursued the same goals as the DigitalDesk. InteractiveDESK [AMK95] used physical objects as links to electronic files. An object placed on the front-projected desk display and identified with a camera above would invoke a menu listing the files associated with that object. The project was continued by supporting hyperlinks from paper documents using a pen with a camera on it that would use OCR on the text under the tip [AAH97]. EnhancedDesk [KK98, KSK01] used a projector to create a desk display, and used a pan-tilt infra-red camera and a new computer vision algorithm to successfully track the user's fingertips, allowing her to interact with the projected information with unadorned hands. Aliakseyeu has added infrared reflecting tags to sheets of paper [Ali02] to track them on a horizontal work surface so they can be augmented with projected imagery. Tele-Graffiti [TSB$^+$01, TSB02] locates a piece of paper mounted on a black clipboard and projects graphics down onto it. Two users can collaborate remotely: each one writes on his sheet of paper with a marker pen, the computer vision system generates *hand-over-paper* events to ensure it gets a snapshot of the paper when it is not obscured, and the image

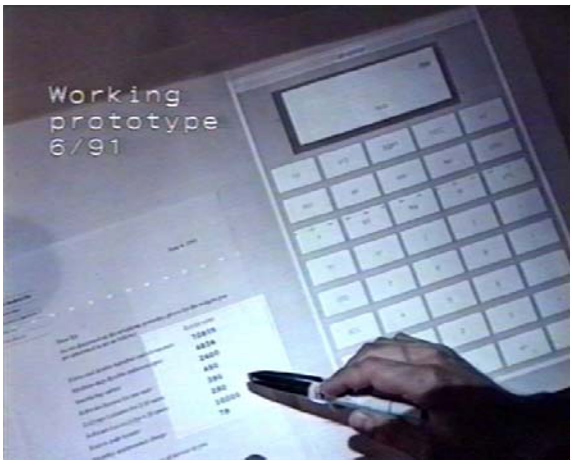


Figure 2.15: The DigitalDesk [Wel94]. Here the user has placed a paper document at a particular position on the desk at which a video camera is pointed. He can copy figures from the paper document to the calculator program on the right to save effort and avoid transcription errors.

is sent to the other user so the two can create a drawing together. It is similar to the DoubleDigitalDesk described below in Section 2.5.2. The Tele-Graffiti system includes hand tracking so the user can select buttons projected onto the desk, and it incorporates a novel software technique for measuring the focus, white balance, hue, and saturation of the camera so that it can be calibrated automatically. LivePaper [RR01] augments multiple sheets of paper placed on a desk with projected graphics (Figure 2.16). A bitmap of each sheet is extracted from the camera image, thresholded, and used to identify the sheet without the use of glyphs by comparing it to the appearance of known sheets. Sheets can be augmented with *literal functionality* or *magical functionality*. Literal functionality extends the standard properties of a sheet of paper by adding graphics that follow the position and orientation of the paper, such as a picture added to a page. Magical functionality remains axis-aligned when the paper is moved, and provides features that normal paper does not have, like buttons that cause music to play.

In contrast to the systems described above that add two-dimensional graphics to paper using front-projection, some systems have added three-dimensional graphics using a transparent head-mounted display. MagicBook [BKP01] recognizes glyphs on the pages of a book and adds a 3D model with which the user can interact. The book acts as an index to a set of virtual worlds. Recently a system was made [RBW01] that arranges the windows of standard application programs on a virtual screen so that they do not overlap, then identifies surfaces in the scene using glyphs, and texture-maps the windows onto the locations of those surfaces in the user's head-mounted display. This turns any surfaces in the user's office into extra screen space. The monitor can be used as the normal focus of attention, with relatively static information available for perusal on the lower-resolution augmented surfaces.

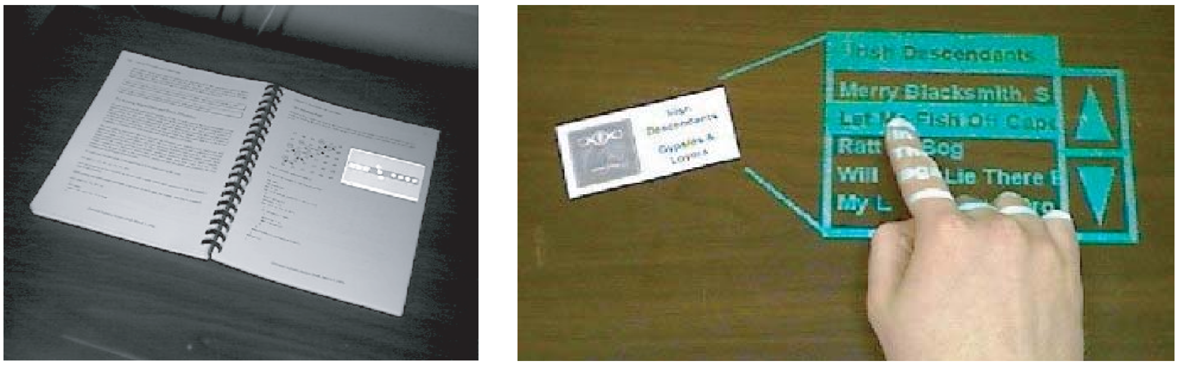


Figure 2.16: The LivePaper system [RR01]. A book (left) is augmented with projected graphics to add *literal functionality* that augments the pictures on the page. A card (right) invokes an associated menu to play songs, which demonstrates *magical functionality* that is not related to the normal physical properties of the card.

Guimbretière [Gui03] has made a system to link paper more closely into the editing cycle—it is easy to output a document to paper, but annotations made on the hard copy cannot normally be processed electronically. A document is held in a digital format, like PDF, and is printed on special paper so it can be annotated with an Anoto [An] pen. The strokes made when the document is marked up are stored by the pen and added to the electronic version of the document so that the user can proof read documents on paper—which is easier than viewing them on a conventional monitor—and then view the annotations on the computer screen after they have been automatically added to the electronic document ready for the next editing stage.

The systems described in this section have used the continued prevalence and unsurpassed affordances of paper to justify hardware and software combinations that aim to include paper in the electronic domain, but I believe that the long-term aim should be to develop holistic approaches to replicating the work practices that currently require paper. As explained in Section 1.2, hardware for electronic books is becoming available, but their adoption now relies on the provision of a distribution infrastructure. Similarly, a suitable interface will be necessary to give electronic documents many of the affordances of physical paper to complement their other features such as the computer's complete control over the data, and the possibility of remote collaboration where both participants interact symmetrically with the document.

2.4.2 Simulating paper

In 1945 Vannevar Bush described his vision of a device called the Memex [Bus45] that could photograph documents from paper, store many of them internally, retrieve one at the touch of a button, and form links between them (Figure 2.17). The fast and simple retrieval of documents was to be a big advantage over a library of paper, and the links would store the process a person had gone through when researching a topic and thus would be an invaluable aid to them or someone else following their work. Various projects have explored the generation of a hyperlink structure at least semi-automatically. Microcosm [FHHD90] stores such a structure as metadata, separate to the original documents, so that existing documents can be linked without modifying the original data, and existing links can be applied to new documents. Gemmel *et al.* [GBL$^+$02] observe that personal computers will soon have so much storage space that deleting anything but large video files will not be worth the effort involved. All of a user's files such as images and text can be amassed over time, but then the challenge is finding the right files for a particular purpose. The researchers are working on the information retrieval side of the Memex vision—linking files to each other to form a graph, and visualizing collections of related information such as images and text.

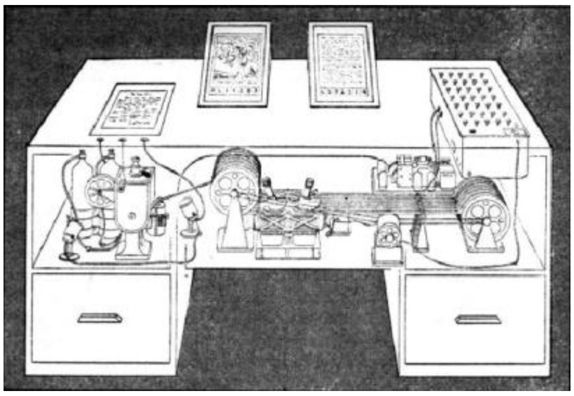


Figure 2.17: Vannevar Bush imagined a device called the Memex [Bus45] in the form of a desk that would instantly bring files and material on any subject to the operator's fingertips. Slanting translucent viewing screens magnify supermicrofilm filed by code numbers. On the left is a mechanism that automatically photographs longhand notes, pictures, and letters, then files them in the desk for future reference.

Interface features have been created that are modelled directly on the physical properties of paper. Some were mentioned briefly in Section 1.2: allowing windows to be rotated through arbitrary angles and causing them to turn as they are dragged [BL01] (Figure 2.18(a)); peeling back windows to reveal those underneath (Figure 2.18(b)); leafing through folders by pulling out the name tabs (Figure 2.18(c)); and adding dog-ears to pages [HS00] so that they become more prominent when flicking through the document (Figure 2.18(d)). Kramer's Translucent Patches [Kra94] are based on the yellow translucent paper that architects overlay on plans. The electronic version consists of translucent non-rectangular patches that are annotated with pen input.

Other projects aim to capture some styles of work that are generally used with physical rather than electronic documents. XLibris [SGP98] uses a portable computer with an LCD panel and pen input to support active reading. Thumbnails of pages other than the currently selected one are displayed around the edges of the screen to facilitate navigation. The shadow of a document thumbnail indicates how many pages the document has. The user can make *clippings*, which are like cuttings from a newspaper and can be sorted on creation time, page number, or the ink colour of the annotations on them, and when the user annotates a page the words, phrases, and passages that have implicitly been selected are used to perform automatic searches whose results are displayed in the margin to facilitate serendipitous retrieval. Dynomite [WSS97], a dynamically organized ink and audio notebook, simulates a physical notebook. Pages are automatically dated, different colours of ink can be used for annotations, and each colour can be toggled between being hidden and visible. The system records the user's voice around the time when something is written and the audio can be played back by selecting the annotation.

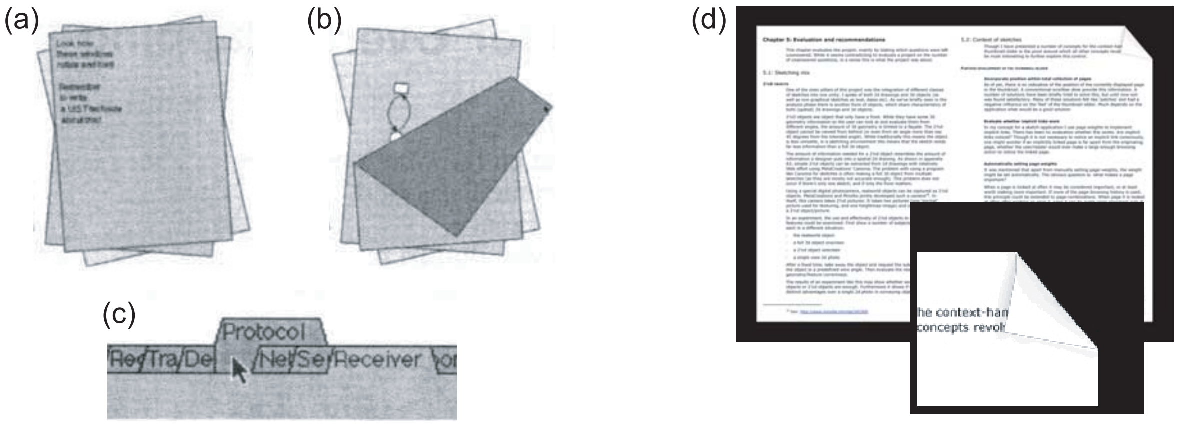


Figure 2.18: Simulating the properties of physical paper in electronic documents by (a) allowing windows to end up at different angles, (b) peeling back windows, (c) leafing through folders [BL01], and (d) having dog-eared pages [HS00]

Data Mountain [RCL98] is a graphical replacement for the menu of favourite web sites on a web browser. It aims to take advantage of human spatial memory by putting thumbnail images of the web pages in a three-dimensional arrangement rather than in a menu, and experiments showed that it reduced subjects' reaction times and error rates compared to the standard menu. Although the 3D layout of Data Mountain is appealing, studies have shown that the 3D feature of the interface has no significant effect on retrieval times for web pages [CK01], which is similar to results showing that 3D visualizations of graphs do not improve over 2D ones. In fact, in a further study [CM02], as the use of 3D increased in such an interface on the computer and using physical cards, retrieval time increased and subjects' assessment of the effectiveness of the interface decreased. The researchers state 'Spatial memory clearly provides an effective aid to information retrieval, but we are skeptical of the role that 3D plays in aiding rapid retrieval of data items from static-perspective spatial organizations.' Some users of Data Mountain wanted to group thumbnails together—a method for piling the items, like that described in Section 7.1, would allow this.

2.4.3 Tangible user interfaces

Physical objects can be augmented with computational properties to get a natural tactile experience when interacting with electronic information which is more like the way people interact with the real world. The Brightboard [SFR96, SF96] was a normal whiteboard augmented with extra abilities via a camera. There was no visual interface between the user and the data—instead the user would write appropriate marks on the board and receive auditory feedback when an action was performed in response. Movement in front of the board triggered the capture and thresholding of an image of the whiteboard, from which marks were recognized and their configuration passed to a Prolog program that responded by running various scripts. By writing particular marks the user could save, print, fax, or e-mail an image from the board. The Tangible Bits project [IU97] concerns tangible user interfaces, and its main focus is the metaDESK [UI97] which aims to physically instantiate many of the metaphorical devices the conventional graphical user interface has popularized. One of its applications is Tangible Geospace, in which the placement of a physical icon of a building, known as a *phicon*, can be moved around on the desk surface causing the back-projected display of a map to move accordingly (Figure 2.19 (left)). Two phicons can be used together to scale and rotate the map. The phicons are tracked by infrared cameras under the desk surface and are illuminated from below by infrared emitters. An alternative view of any part of the map is displayed when a device called the *passive lens* is moved over the top. This device is a physical version of the magic lenses described in Section 2.4.4 below, and is simply a circular object with a large hole in the middle, that has a tracking device attached. The metaDESK also has an *active lens*, which is an arm-mounted flat-panel display that allows the user to view buildings from the map in 3D by moving the panel into the appropriate place as if he is looking through the panel to see the buildings behind.

A luminous tangible workbench for urban planning and design called Urp [UI99] has wire-frame buildings that the architect or planner positions on a horizontal surface to experiment with the plan of a site. The buildings are tracked by a camera above, then graphics projected from above show ramifications of the layout the user has chosen—for instance, shadows at different times of day or airflow around the buildings. Architects that viewed the system were enthusiastic about the possibility for rapid prototyping and for presentations of ideas to clients. The Sensetable [PIHP01] is a large tablet that can

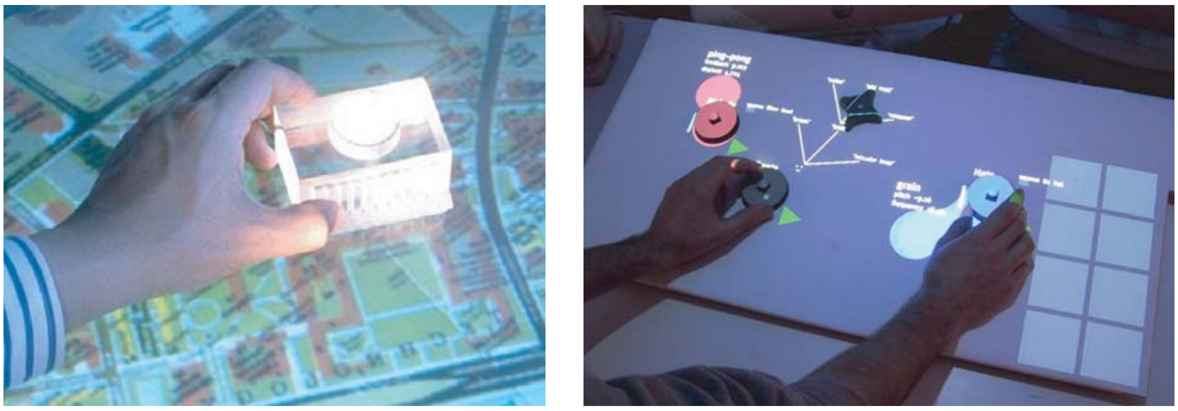


Figure 2.19: The metaDESK [UI97] and Sensetable [PIHP01] projects from the Tangible Media Group at MIT. They are examples of tangible user interfaces where physical objects allow many parameters of a system to be controlled with minimal effort spent switching between parameters.

track many pucks at once. It has been used for chemistry simulations where the pucks represent atoms and molecules that can be brought together. Audiopad [PRI02] has taken the Sensetable further with smaller pucks, and a continuous surface rather than multiple smaller tablets placed adjacent to one another. It is used to produce music as shown in Figure 2.19 (right) where the volumes of various tracks are controlled by rotating the corresponding pucks.

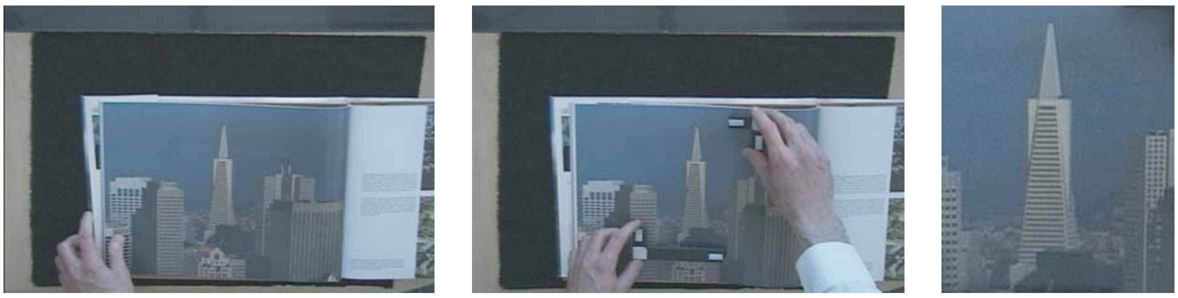


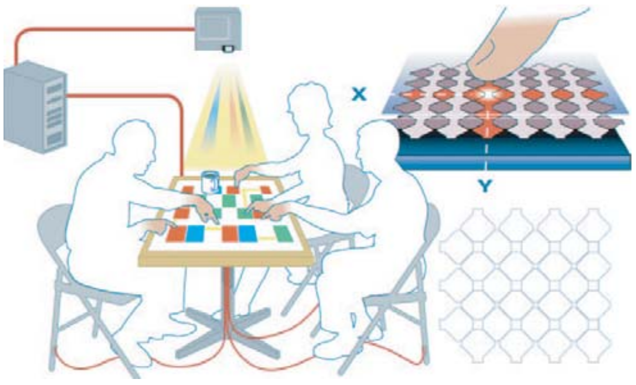
Figure 2.20: FlowScan [Gui02] allows pictures to be added to the Stanford Interactive Mural. The user puts a picture in position (left), places crop markers on it (centre), then a camera automatically samples the picture (right) and places a copy on the display wall.

A brainstorming system implemented on the Interactive Mural allows an image to be added to the set of graphical items on the wall display using a tangible interface called FlowScan [Gui02, Section 5.3]. Instead of using a flatbed scanner the user places physical crop markers on a picture, and an overhead camera adds the image to the wall display (Figure 2.20). This lets the user easily bring information that has been found on paper into the computer-supported session without breaking the flow of ideas.

2.4.4 Input methods

There are various possibilities for input to computer systems other than the conventional keyboard and mouse, including tablets with puck and stylus input which have been available for decades but are seldom used outside specialized domains. Sukthankar *et al.* [SSM00] have used laser pointing from a distance to control a projector for presentations. The system uses a camera to locate the laser dot, generates events to make the mouse follow it, and generates a mouse click when the dot dwells in a small area for a certain period. The creators of LumiPoint [DC00, DC02] believe that as a display gets bigger the number of input devices should scale accordingly to allow multiple people to use it. Their system tracks multiple laser pointers from behind a back-projected screen with only 20 ms latency, and allows the pens to be used close to the screen or at a distance. It avoids confusing the different beams by predicting their future locations based on their past movement.

The MIT Media Lab has experimented with various technologies for sensing touch on large vertical surfaces including transmitting an electric field through the user and sensing it with receivers at the corners of the display, scanning in front of the surface with a laser, and passing light into one edge of a plexiglass sheet that the user touches then detecting it with photodiodes on the opposite edge [Par02]. DiamondTouch [DL01, LD02] tracks the hands of up to two users on a horizontal surface by transmitting an electrical signal through the chairs they are sitting on, through the users, and into a grid of antennae in the table (Figure 2.21 (left)). The device has been incorporated into a front-projected photo-browsing interface [SLV03] that can be used as coffee table and displays photographs on a virtual rotating circular turntable. SmartSkin [Rek02] is a similar system that uses a mesh of copper wires under the surface of a table to detect the capacitance of users' hands. It can track multiple hands on a table-sized system (Figure 2.21 (right)) and can estimate the distance of the hands from the surface. Both DiamondTouch and SmartSkin are currently research prototypes and are not available as commercial products.



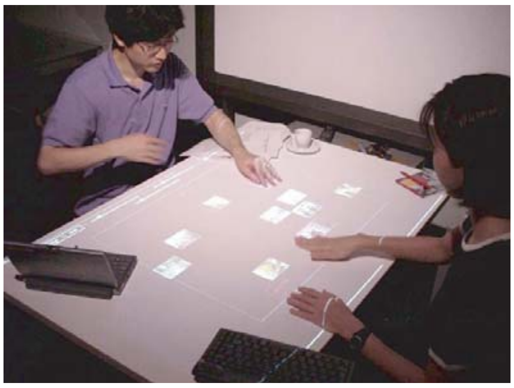


Figure 2.21: Prototype electrical touch sensing technologies: (left) DiamondTouch [LD02]; and (right) SmartSkin [Rek02]. Both devices use a grid of wires embedded in the surface to track multiple hands.

A system called EnhancedDesk uses computer vision to track the two hands of the user as they move over the horizontal surface, and avoids the problems of varying lighting conditions and a complex background by using an infrared camera tuned to the temperature of human skin. The image is thresholded and circular templates are used to find the fingertips [SKK00]. Drawing is accomplished using two-handed gestures [OSK02, KXN+02] where, for instance, the left hand specifies the centre of a circle and the distance between the two hands specifies the radius.

An important feature of physical media like paper is the passive haptic feedback that the subject gets, which is of three types: proprioception, the perception of stimuli produced within the body; kinæsthesis, the natural sense of bodily movements; and finally the feel of the input devices. Another important feature is the way that physical media can be used with two hands, which does not just save time because the hands work in parallel, but also changes the way the person thinks about the task [HPPK98].

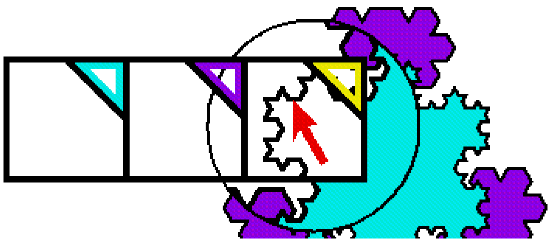


Figure 2.22: Toolglasses and magic lenses [BSP+93]. The non-dominant hand can position tools to modify items or change the way they are displayed, and the dominant hand can *click through* the tools to change the items underneath. In the figure the circular magic lens magnifies the object underneath, and the toolglass allows the pointer controlled by the dominant hand to click-through and change the colour of the item.

Buxton [Bux94] states that tasks performed by the hands can be divided into foreground and background tasks, and the use of two hands allows these to be overlapped. Also, from the display perspective the conventional user interface is space multiplexed, with arrays of buttons to show the options that are available, but from the control perspective it is time multiplexed, because tools are selected one at a time. Two-handed input allows the acquisition period for each tool to be overlapped with a task performed by the other hand, which saves time. Experimental data suggest that performance increases can be gained by splitting the sub-tasks of compound continuous tasks between the two hands [BM86]. Two important advantages of bimanual manipulation are [LZB98]: the manual benefits from increased time-motion efficiency due to twice as many degrees of freedom being available to the user, and the cognitive benefits as a result of reducing the load of composing and visualizing the task at an unnaturally low level imposed by conventional unimanual techniques. The non-dominant hand may not be able to perform work as detailed as that of the dominant hand, but as explained in Section 1.3 the roles of the hands are complementary. An experiment by Buxton *et al.* [KMB93] on selecting targets with a mouse, a track ball, and a pen which were used by both the dominant and non-dominant hands, showed that for large targets and large distances the non-dominant hand performs just as well, so it can usefully be assigned imprecise tasks like scrolling a document.

Toolglasses and *magic lenses* [BSP$^+$93] are user interface tools for use with bimanual input. A toolglass is a semi-transparent widget that is coarsely positioned with the non-dominant hand to make a context for the dominant hand to click-through and alter the object underneath. A magic lens is a viewing filter that changes the representation of the object underneath (Figure 2.22). These tools are brought by the non-dominant hand to the area to which the user is attending so he can stay focussed on his work, and they do not require any permanent screen space so they can be adapted to a wide range of display sizes. These tools were used in a graphics application with two pucks for bimanual input [KFBB97] whose design goals were: maximize the screen space used for artwork, avoid forcing the user to divert his visual attention from the artwork, and increase degrees of manipulation and comfort of input. A group at the University of Aarhus in Denmark has combined floating palettes, toolglasses, hierarchical marking menus, a novel window model, and some new interaction techniques into a comprehensive tool for editing coloured petri nets [BLMA$^+$01, CPN]. They have provided bimanual input with a mouse for the dominant hand and a trackball for the non-dominant hand. In Rekimoto's system described above in Section 2.2.1 in which a tracked PDA was used in conjunction with a large screen, the PDA could be used like a toolglass by placing it near an item on the board and selecting an option—effectively clicking through the PDA to alter an attribute of the item underneath.

2.5 Collaboration

Synchronous interaction between people can either be between copresent people or distributed people—the two conditions on the left of Figure 1.5 on page 14. Systems that support those two types of activity are described below. The Escritoire provides synchronous distributed interaction that I have designed to be as if both participants are working at the same physical desk with real sheets of paper.

2.5.1 Copresent

Single Display Groupware [SBD99] is the name given to systems where multiple users are placed together and use the same computer. With the Two-user Responsive Workbench [ABM$^+$97], a two-person version of the Responsive Workbench [KBF$^+$95], the participants share a 3D display that uses frame interleaving to present each one with a stereo image pair (Figure 2.23). Independent views are possible for the two participants, so they can have different representations of an object that match their different roles

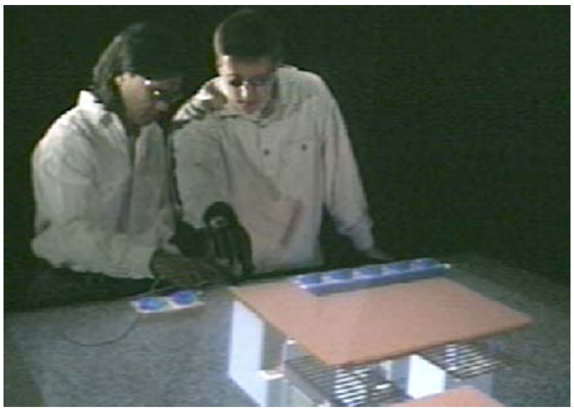


Figure 2.23: The Two-user Responsive Workbench [ABM+97]. Each of two users is presented with a stereo image so they can work together on the same 3D model. The users wear shutter glasses and their heads are tracked so the model can be rendered from their separate view-points.

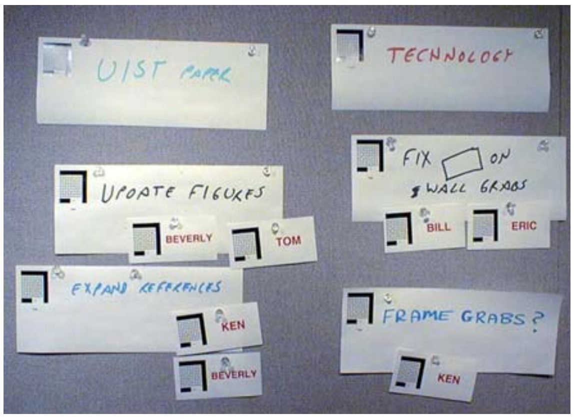


Figure 2.24: The Project Task Wall application [MSvM+99]. Pieces of paper are identified by their glyphs, and their arrangement is interpreted as an assignment of people to tasks. The assignment is made available via a web page for remote users who cannot see the physical board.

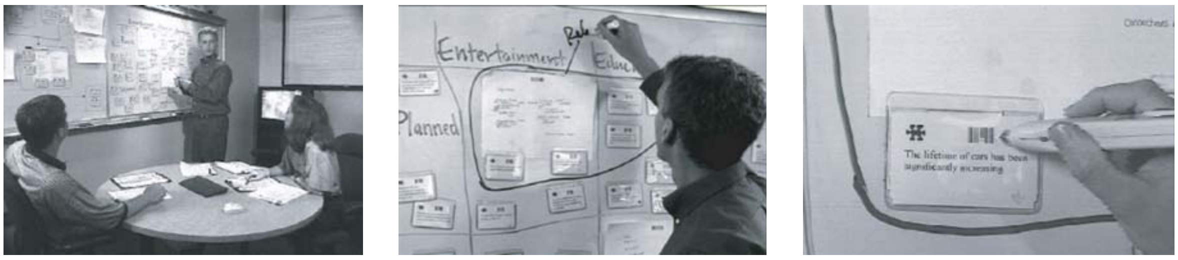


Figure 2.25: Insight Lab [LJM98] uses paper cards to represent text, images, and video clips. They are arranged during a discussion, and the related materials can readily be viewed by scanning the bar-code on any card.

in the task, different levels of detail that depend on their separate foci of attention, or private workspaces.

Meeting support is a major focus of groupware technology, and documents are usually essential to a meeting: people will bring prepared documents to a meeting, and they will make hand-written notes that may form the basis of documents that will be used in subsequent meetings. DOLPHIN [SGHH94] was a meeting support system for working on formal or informal information, and transforming between the two. It stores content as *scribbles*, text, and images, and arranges it in a hypermedia document model with nodes and links. It was designed to support interactions of all four types shown in Figure 1.5, that is, both face-to-face and distributed collaboration, and both concurrent and asynchronous activity: during a meeting participants can be in a single room and use the Xerox Liveboards (Section 2.2.1 above) that are provided or they can work from their desks via networked computers, and a user can work on a document individually then produce it from a private workspace during the meeting. A system called Knight [DHT00] allows software designers to work together in a single location on an object-oriented design using a whiteboard-style interface displayed on a SMART Board [ST]. The Collaborage system at PARC [MSvM+99] uses computer vision to make a wall into a computerized interface that is tangible and provides simple audio feedback, like the BrightBoard (Section 2.4.3 above). Collaborage is in the same-place different-time category of Figure 1.5. One application is the Project Task Wall where cards representing people and tasks are arranged on the wall (Figure 2.24), glyphs on the cards are used to sense their positions, and a version of the information is made available on a web page. The Insight Lab [LJM98] developed at Andersen Consulting, now Accenture, is another tangible user interface based on arranging paper on the walls of a room. Participants analysing observational data such as pieces of evidence—quotes, images, video clips—and keywords put cards that represent the data into groups based on patterns and hypotheses. The pieces of evidence can be reviewed on large screens by scanning bar-codes on the cards (Figure 2.25). Once the team has discussed and organized the items it can store any structure decided upon by scanning the bar-codes to group items.

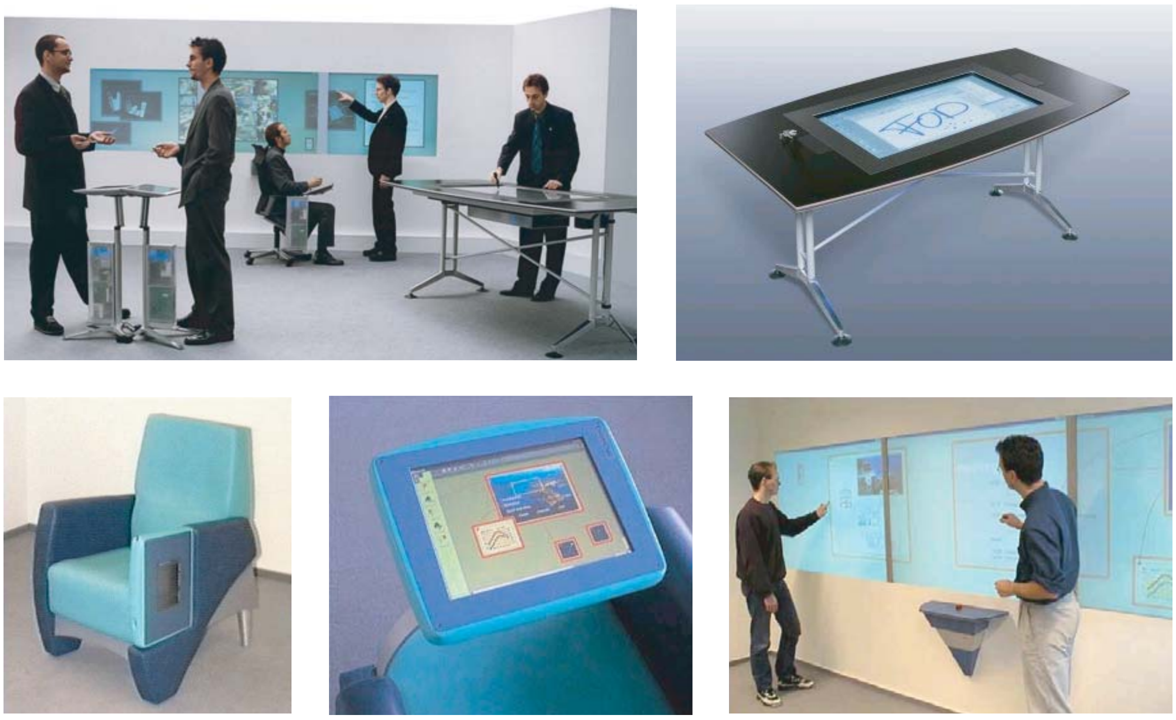


Figure 2.26: Roomware [PT02] components allow collaboration around displays of different types with varied affordances (top left). The *InteracTable* (top right) allows many users to gather around, the *CommChair* (bottom left and bottom centre) provides a private workspace, and the *DynaWall* (bottom right) presents a large workspace on which users can work separately or collaboratively

The i-LAND project [SGH+99] was based on a vision of workspaces of the future that support co-operative work of dynamic teams, and has been continued with Roomware [SPMT+02, PT02]. The position of the desktop computer as the primary point of access to information and as a bottleneck due to limited screen space and complex handling of windows are used as motivations for integration of architectural and informational spaces. Roomware components called the *InteracTable*, *CommChair*, and *DynaWall* have been developed, and are shown in Figure 2.26. The InteracTable is a horizontal 65×85 cm display that allows drawing and gestures with a finger or pen, and text input via a wireless keyboard. Because up to six people can stand around it at once, new interface techniques are needed that have no predefined orientation. The DynaWall is a 4.5×1.1 m touch-sensitive display that supports some novel interaction techniques, such as being able to throw an item across the display to a user on the other side who can catch it. The CommChairs contain wireless network devices, independent power supplies, and pen-based computers that provide a workspace that can be used for private work or to remotely add annotations to the DynaWall.

In many conventional tasks centred around a computer the participants have very different roles, for example doctor and patient, or salesperson and customer. These systems can place a barrier between the participants because information is tailored for, and displayed to, one person—the doctor or salesperson—who then has to convey it verbally to the other. The situation can be improved by using large displays and presenting the information in a form that both people can understand and refer to during the conversation, transforming the computer from being a barrier to being the common focus of attention [RRHT03].

2.5.2 Distributed

As I explained in Section 1.4, distributed collaboration between people can create a person space or a task space. I will concentrate on systems where the participants work in a task space, rather than on conventional videoconferencing systems that follow the the *talking heads* model.

Around 1970 Myron Krueger made VIDEODESK [Kru93]. Two users sat at desks in different locations, each saw a screen showing text and graphical information, and when a user moved his hands over the desk they were visible on his and his partner's screens. The physically separated users could talk about the information while gesturing with their hands. VideoDraw [TM91] allowed separated users to draw with standard marker pens and gesture to each other with their hands. It was a simple analogue system that consisted of two CRT screens with sheets of glass over them, connected to two television cameras pointing at the screens (Figure 2.27). Combining the task space of VideoDraw with a person space resulted in ClearBoard [IKG92] which used the metaphor of drawing on a glass surface between two people. The first version was created with video hardware and half-mirrors. The image of each user was flipped horizontally so they had a common drawing orientation and each one could read text written by the other. Gaze awareness allowed each user to see what the other was looking at, which enhanced the feeling of copresence. The second version of the system used a digitizer and pen input so drawings could be saved, loaded, and imported, and so one user could alter marks made by the other.

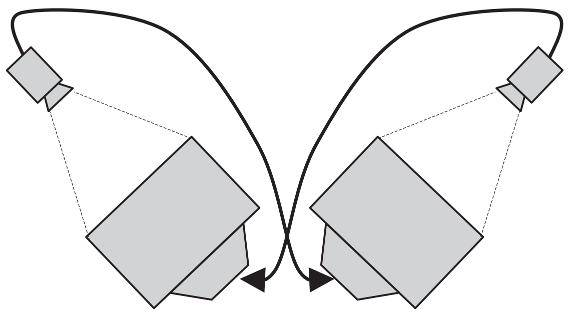


Figure 2.27: VideoDraw [TM91] explored collaboration in a task space. Each of two television screens showed the image generated by a camera pointing at the other. Users could work on a picture by drawing on their screens and gesturing with their hands.

GroupSketch and GroupDraw [GRWB92b] are early examples of task spaces between distributed users. They were created with GroupKit [RG92], a toolkit for making synchronous distributed groupware that supported freehand drawing for annotation, and multiple cursors over the application area for gesturing between users. A fisheye text editor was created to allow multiple users to concurrently edit a large document while having awareness of each others actions [Gre96]. The whole document is displayed on the screen in a very small font, and the fisheye effect increases the font sizes around the focal region of the local user and to a lesser degree around those of the remote users. Colour is used to indicate which user is working in each region.

The DigitalDesk, described above in Section 2.4.1, was augmented to create the DoubleDigitalDesk [Wel94] which combines views from video cameras above two desks so that users can collaborate remotely using paper documents. The video images from each desk, which may include the user's hands, are scaled, thresholded, and sent to the other desk (Figure 2.28). Multiple distributed input devices were implemented for the DoubleDigitalDesk, and also for the Multi-Device Multi-User Multi Editor (MMM) [BF91, Fre93] a multi-user text editor with menus and a hierarchy of windows that can be used simultaneously by multiple users, each with her own mouse. In MMM each user picks a mouse to use then clicks in her *home area* to inform the system of who is using it. Interference between users is reduced because fine-grained sharing allows users to simultaneously manipulate a single window or string of text with confidence that their actions will not hinder those of other users.

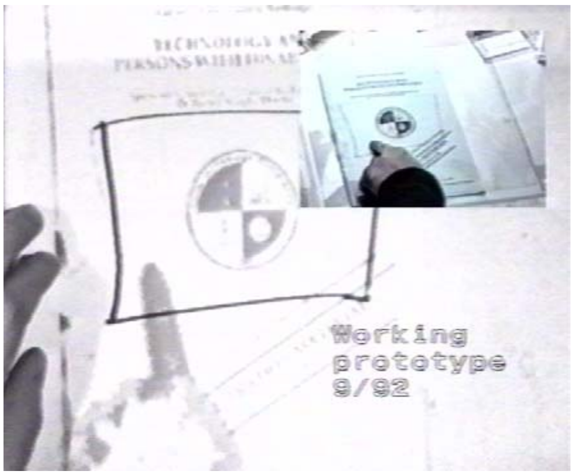


Figure 2.28: The DoubleDigitalDesk [Wel94] allows two users in different locations to work on the same paper document. Here the remote user (inset) is pointing at the sheet of paper and an image of his hand is displayed for the local user.

The use of real paper in a collaborative system has been continued with a system called the Designer's Outpost [KE02, EKLL03] in which sticky notes are placed on a large rear-projected display (Figure 2.29). When the camera in front of the display recognizes a note, a shadow is displayed around it to indicate that fact, and the image of it is sent to the remote display. When a local user moves the note its new position is reflected on the remote display, but a remote user cannot move the physical note. Instead he moves the graphical representation of it, and the note is then highlighted in red, indicating that it is *transactionally inconsistent* and that the local user should move it to the new position. Users can select *transient ink* from a pie menu to make gestural marks during conversation. About half of test participants found the transient ink useful. For the Escritoire I have implemented pen traces, which are described in Section 7.2, that are always on rather than needing to be selected from a menu, and which fade in a way that makes the movement of the pens apparent. They

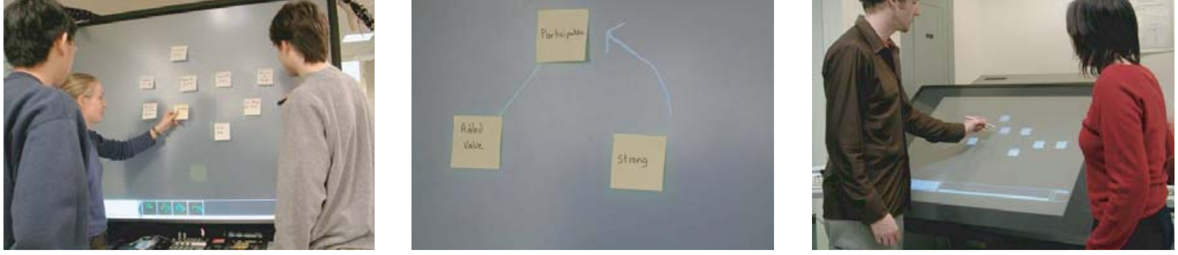


Figure 2.29: The Designer's Outpost system [EKLL03] is designed to allow participants in different locations to work with the same arrangement of paper notes. Those at one end can manipulate the physical notes. A remote user can move a note's virtual counterpart which then indicates an inconsistency that the local user must correct by moving the physical note to its new position.

also have the advantage that because they are not consciously activated they give continual feedback on the actions of a remote user which enhances the awareness between users, and I believe they are a much better solution for gesturing. The use of paper in a remote system necessitates concepts such as transactional inconsistency and the inconvenience of correcting such inconsistencies so I have used virtual paper instead, that can be manipulated by local and remote participants alike.

2.6 Summary

Researchers have used visualization techniques like the sequential display of workspaces in Rooms, or the distortion of fisheye views, in attempts to make the most of the limited display area that a conventional monitor screen offers. Level of detail rendering for 3D scenes aims to make the most of the computational power available by rendering parts of the scene differently depending on the resolution that is needed, and it is this technique that most resembles the rendering required on the display of the Escritoire which has the unusual property of non-uniform spatial resolution. The ultimate answer to the limited area of conventional screens is large display devices, which have generally been formed in the style of a whiteboard. Various human factors issues arise with large displays that make their design requirements different to those of conventional monitor screens. For instance, the large area of such a display should not be regarded as perceptually homogeneous for the user, because people will naturally arrange their work in a smaller area which is supported by the visual periphery provided by the large display. Digital projectors are currently the only commercially available technology for creating scalable large displays, and various research groups have assembled arrays of projectors driven by clusters of rendering and compute nodes. Much effort has recently been focussed on geometric and photometric calibration of such displays, and projectors have also been used for augmenting physical objects with projected imagery.

In Section 1.2 I explained that paper is still widely used because it has affordances that are not covered by the conventional computer interface. This can be addressed in two ways: by augmenting physical paper with computational abilities, as did the DigitalDesk and similar systems after it, or by simulating the properties of paper. I believe the latter is the long-term solution, especially when remote collaboration is required: a graphical interface can be created with features such as those in Figure 2.18, and systems like Satchel (Section 1.2) can provide the ability, for instance, to hand a document from one person to another when they meet at any location. Tangible user interfaces provide a method of interaction that makes use of the skill and range of motion that humans display when working with physical media, and the DiamondTouch and SmartSkin systems demonstrate prototype desk-sized sensing systems that are operated with unadorned hands. Bimanual input has been shown to have manual and cognitive benefits, especially when the difference between the dominant and non-dominant hands is considered, so it has been used as an integral part of the Escritoire's interface.

Finally this chapter describes some collaborative systems for users in the same physical location, and in different physical locations. The Designer's Outpost illustrates the problem of trying to collaborate remotely using physical paper, which can only exist at one end of the link between the parties. Section 1.4 introduces the concepts of task space and person space, and I believe that for many domains a system that provides task space between the participants will be much more useful, so my description concentrates on such systems, and I have designed the Escritoire to provide a task space to complement the person space of a standard videoconference.

Chapter 3

The Escritoire

To address the lack of space in conventional user interfaces and the disparity between their affordances and those of a desk with physical sheets of paper I have created a personal projected display called the *Escritoire* [AR02, AR03c]. The system uses two overlapping projectors to form a horizontal desk display that is large enough to accommodate many items such as documents and images, and also has a high-resolution area in which the user can perform detailed work. I have called this combination of projectors a *foveal display*. There are two pens with different functions that allow the user to interact with the items on the desk using both hands.

3.1 Overview

The escritoire presents a horizontal display to the user that is like a real desk, rather than the vertical screen of a conventional personal computer. I believe that this is a more natural format for working with documents. Objects have different affordances depending on features such as their shape and the materials from which they are composed [Nor88], and these affordances determine how the objects are used. Pinhanez *et al.* [PKL$^+$01] found that users responded to projected displays differently depending on the surface on which the display was being projected. The affordances of a display that has the size and orientation of a real desk, and which is controlled by simultaneously using both hands, allow techniques more like those employed with real paper.

Figure 3.1: Projectors are getting smaller and cheaper. This is a Proxima DX3: dimensions 245$\times$195$\times$68 mm, weight 2.3 kg, resolution XGA (1024$\times$768), brightness 1100 lumens, cost under 2000 euros.

Digital projectors have been getting cheaper, brighter, smaller, and lighter, with technological advances such as the Digital Light Processing chip from Texas Instruments

[Yod97]. The Proxima DX3 projector that was used in the first Escritoire weighs just 2.3 kilograms, costs under 2000 euros, and delivers 1024×768 resolution (Figure 3.1). The native resolution of a projector is determined by its LCD or DLP component. If the projector is driven with a video signal of a different resolution it will apply scaling, resulting in image degradation. Currently the LCD and DLP components are only produced in bulk for SVGA (800×600) and XGA (1024×768) resolutions. Projectors with higher resolutions are not mass market products and are consequently prohibitively expensive. To create an interface the size of a desk I have used two projectors to create a *foveal display* which has the high resolution only where it is needed. It avoids the high cost and complex infrastructure of the multi-projector display walls described in Section 2.3.1 and is thus feasible for individual users. The two projectors are driven from a single PC using two graphics cards or one of the dual-head cards that are becoming common. Projector bulbs are typically expected to last 2000 or 3000 hours and cost around 500 euros, which means that they cost 15 to 25 cents per hour to run, and a bulb used for 40 hours each week should last one to one and a half years.

3.2 Wall display

Initially I made a large display by projecting a conventional desktop onto a wall, and creating *wands* with which to control the programs [AR01], as shown in Figure 3.2. The wands are plastic tubes containing Polhemus Fastrak [Pol] magnetic trackers that report their location and orientation in 3D space. Figure 3.3 (left) shows the wands and the device that emits the magnetic waves. Figure 3.3 (right) shows the wiring that allows the state of the buttons embedded in the wands to be sensed via the parallel port of the computer.

The Fastrak system consists of a device that emits magnetic waves, and receivers that sense their location in the magnetic field and report their 3D location and their

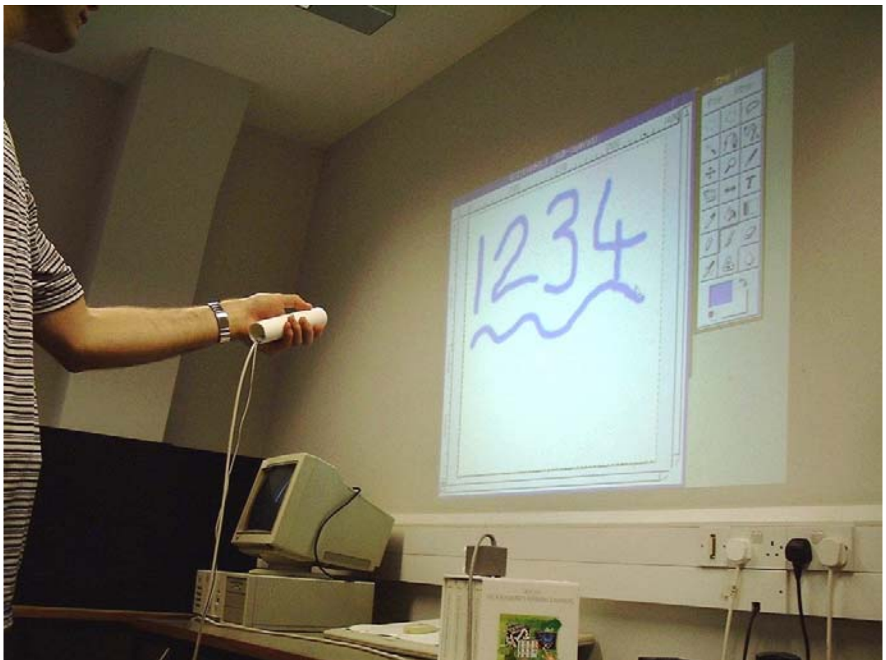


Figure 3.2: Creating a display on the wall: an unmodified drawing program for the X Window System is controlled from a distance by pointing the wand at the wall.

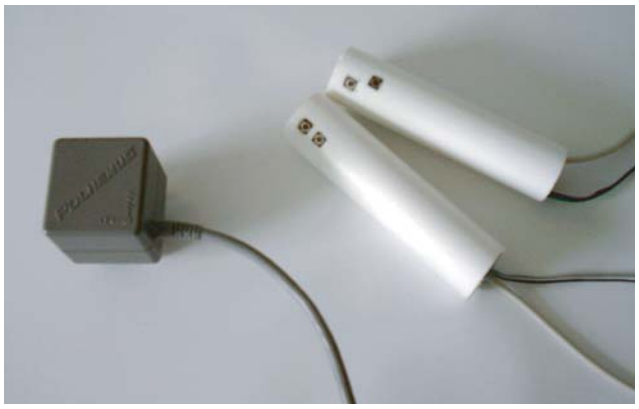
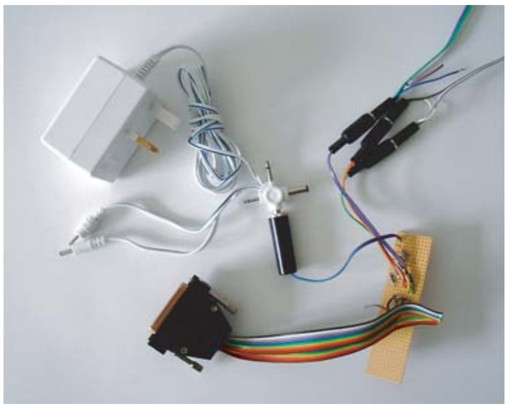



Figure 3.3: *Wands* containing magnetic trackers: (left) the magnetic emitter, and the plastic tubes of the wands that contain the magnetic receivers and buttons; (right) the wiring used to sense the button presses via the computer's parallel port.

orientation as three angles. To use the wands to control the wall display their six-degree-of-freedom data has to be converted to 2D mouse events. I did this in two stages: by firstly getting a 2D position in the plane of the wall, then secondly warping this position to compensate for the distortion of the projected image to get a pixel location on the display.

I determined the position of the wall by placing the wand on it in three separate positions, thus generating three points that define a plane. I implemented two methods for producing a 2D position from the location and orientation of the wand: creating a perpendicular line from the surface to the wand, and intersecting a line through the wand with the plane of the wall (Figure 3.4). The first method uses only the location data from the wand, while the second uses the location and orientation data. They produce the same effect when the wand is touching the display surface, but the first method requires the same large arm movements even when the user is further away from the surface, while the second requires only small changes in the orientation of the wand.

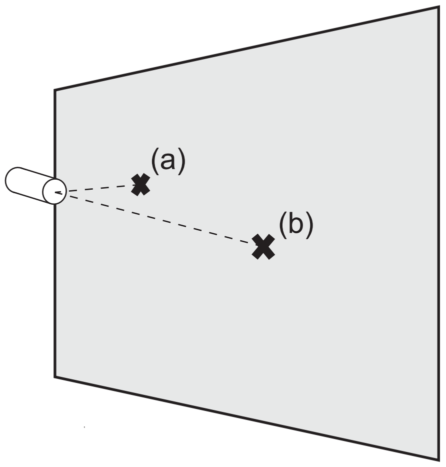


Figure 3.4: Generating a 2D event from the wand's location and orientation: (a) we can project the location orthogonally onto the plane, or (b) we can intersect a line through the wand with the plane.

To compensate for the distortion of the projected image I applied a *four point warping* to the 2D positions in the plane, as used on the DigitalDesk [Wel94], and the corresponding pixel locations were used to generate mouse events.

I used the wall display with a standard drawing program to perform demonstrations for visitors to the Computer Laboratory. Spectators were invited to try out the interface because they could be handed the wand and could then control the display at a distance. It became apparent that the users could not use the perpendicular method (Figure 3.4(a)) for generating events—they found the mapping between wand location and cursor position difficult to understand, and this was probably exacerbated by the need for large arm movements which could make the user self conscious. The other method (Figure 3.4(b)) was readily accepted. Reasonable control was possible even though the method is very sensitive to changes and errors in the orientation of the wand. A cursor to show the user where he is pointing on the display greatly aids hand-eye co-ordination. During a presentation the speaker looks and points with the wand when he wants to affect the display, which naturally focuses the attention of the audience on the right location at the right time. This makes it easy to alternate between speaking to the audience and controlling, for example, a presentation program such as Microsoft PowerPoint.

The four point warping mentioned above warps the input events to match a distorted projection, however, actually fixing the distortion is necessary if multiple projections are to be aligned—Chapter 4 describes a method for doing this. Alternatives to the four point warping for calibrating pen devices with different characteristics are given in Section 5.1. A magnetic tracking device cannot be used near metal objects because they distort the magnetic field. Section 5.2 describes a method for calibrating the wands without placing them on the display surface, which avoids the problem of surfaces containing metal and also allows display surfaces that are out of reach to be calibrated and controlled from a distance.

3.3 Foveal display

A large single-projector XGA display like the one described above in Section 3.2 does not have sufficient resolution to allow an A4-size document to be read. An A0-sized display that is 1024 pixels wide has a resolution of 22 dpi. Researchers using the Liveboard [EBG$^+$92], a 1120×780, 46×32 inch, 24 dpi, interactive whiteboard described in Section 2.2.1, found that, above all, surveyed users would have liked better image quality. The use of a multi-projector display wall like those described in Section 2.3.1 would be undesirable because of the space needed, the complexity involved, and the cost of the computers, network, and projectors. The foveal display achieves a compromise between cost, size, and resolution by providing high resolution only in the area close to the user. It has a *periphery* that is large and can contain many items, and a *fovea* that is small, has high resolution, and can be used for detailed work, with items being chosen from the periphery and brought into the fovea as necessary. Figure 3.5 shows what the display looks like in use and illustrates the difference between the fovea and periphery.

The system is for a single user and is intended to be used in a normal office environment so the precise mechanical calibration that has been used for multi-projector display walls [HJS00, DW] and immersive environments [CNSD93] would be problematic. Complex mountings like the one in Figure 2.11 on page 29 would be undesirable. The use of a curved mirror to allow projection from above onto a sloping drawing board was suggested by Carter [Car93b], but custom optics of this type are unnecessary—Chapter 4 describes how commodity video hardware can be used to warp the projected graphics to achieve the same effect. Because the warping is controlled by software the projectors can be positioned roughly then a short calibration routine is performed in which the user selects some projected targets with the pen. Cameras were used by Wellner [Wel94] and Sukthankar *et al.* [SSM00, SSM01] to locate projected targets, but no cameras were used in the Escritoire so that no reliance had to be placed on the robustness of computer vision techniques that would have to operate in a range of conditions, and no restrictions had to be placed on the visual properties of the desk. In any case, even if a

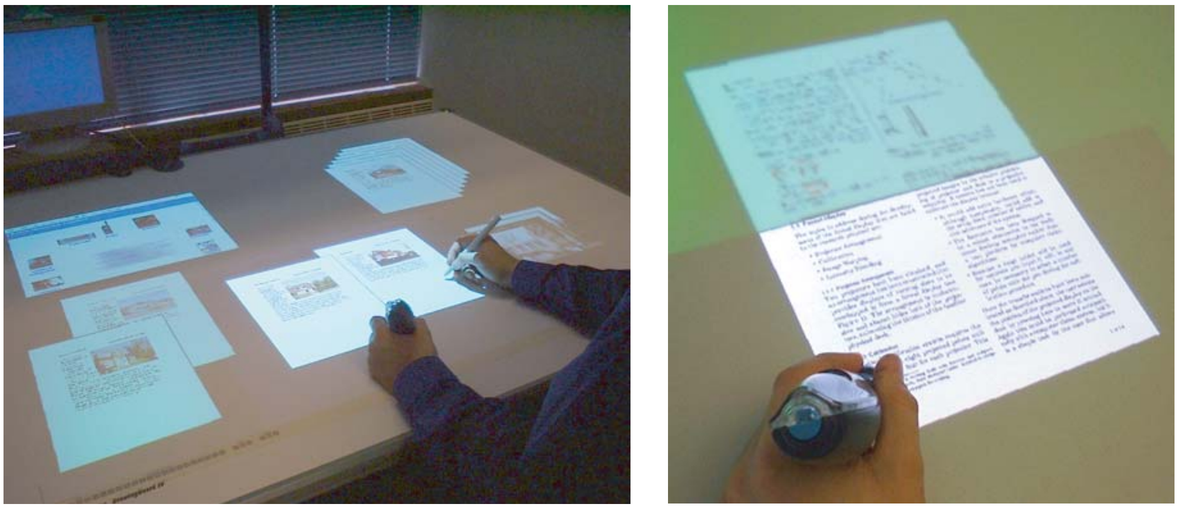


Figure 3.5: The foveal display: (left) the large periphery allows many items to be strewn across the desk and the smaller fovea can display two A4 sheets in detail; (right) a sheet is being dragged with the pen and is halfway into the fovea.

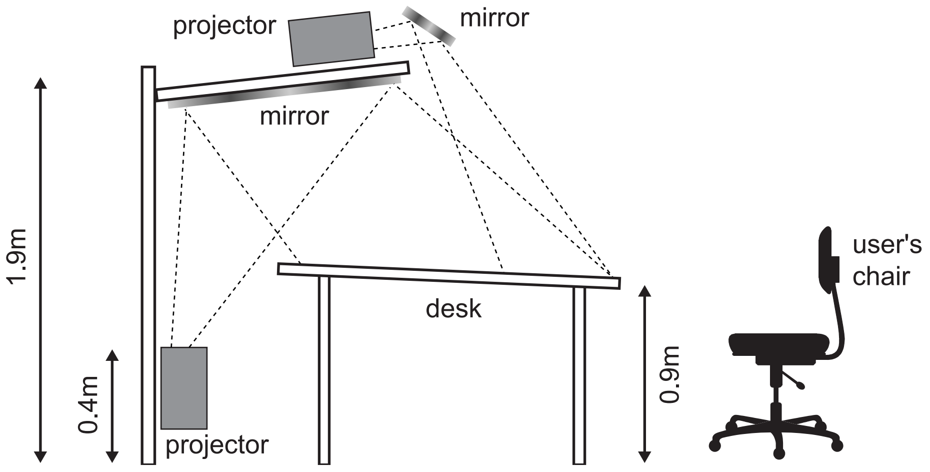


Figure 3.6: The two-projector arrangement of the foveal display. A shelf above the desk holds one projector that reflects its image in a small mirror to make the fovea. Another projector that is mounted behind the desk reflects its image in a large mirror on the bottom of the shelf to make the periphery. The height and angle of the shelf are adjustable. The measurements are approximate.

mapping from desk to projector was obtained completely automatically, it would still be necessary for the user to perform a routine to calibrate the pens to the desk by selecting some known points.

Projectors have zoom lenses but there are still limits on the range of image sizes that can be generated from a particular throw distance. To make a foveal image as small as an A3 sheet of paper a typical portable projector at minimum zoom requires a throw distance of 0.75–1.0m, and to make one as large as an A0 sheet it requires, at maximum zoom, a distance of 1.9–2.4m. This means that the projectors that produce the fovea and periphery of the foveal display cannot simply be placed next to each other with the same throw distance and different zoom settings. The projector arrangement I have used is illustrated in Figure 3.6. Both projectors are out of the way of the user, they have the necessary throw distances, and the total height is within the ceiling height of a typical office. A periphery of A0 size that is 1024 pixels across has a resolution of 22 dpi. A

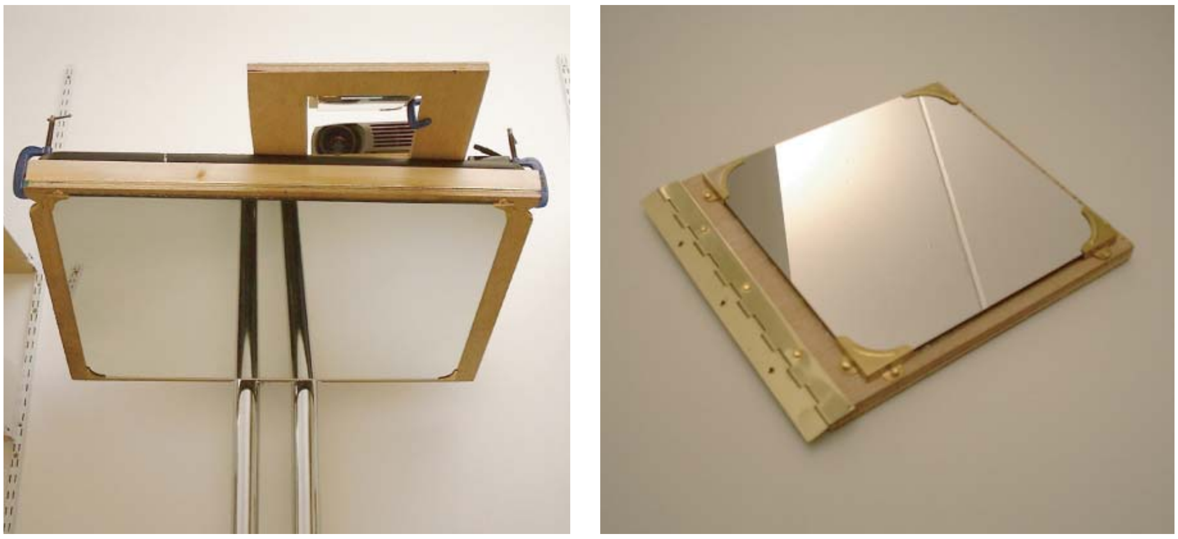


Figure 3.7: In the first version of the Escritoire hardware the projector for the fovea (left) was placed on top of the shelf above the desk that holds the large mirror, and had a smaller mirror (right) that reflected it's image down onto the desk surface.

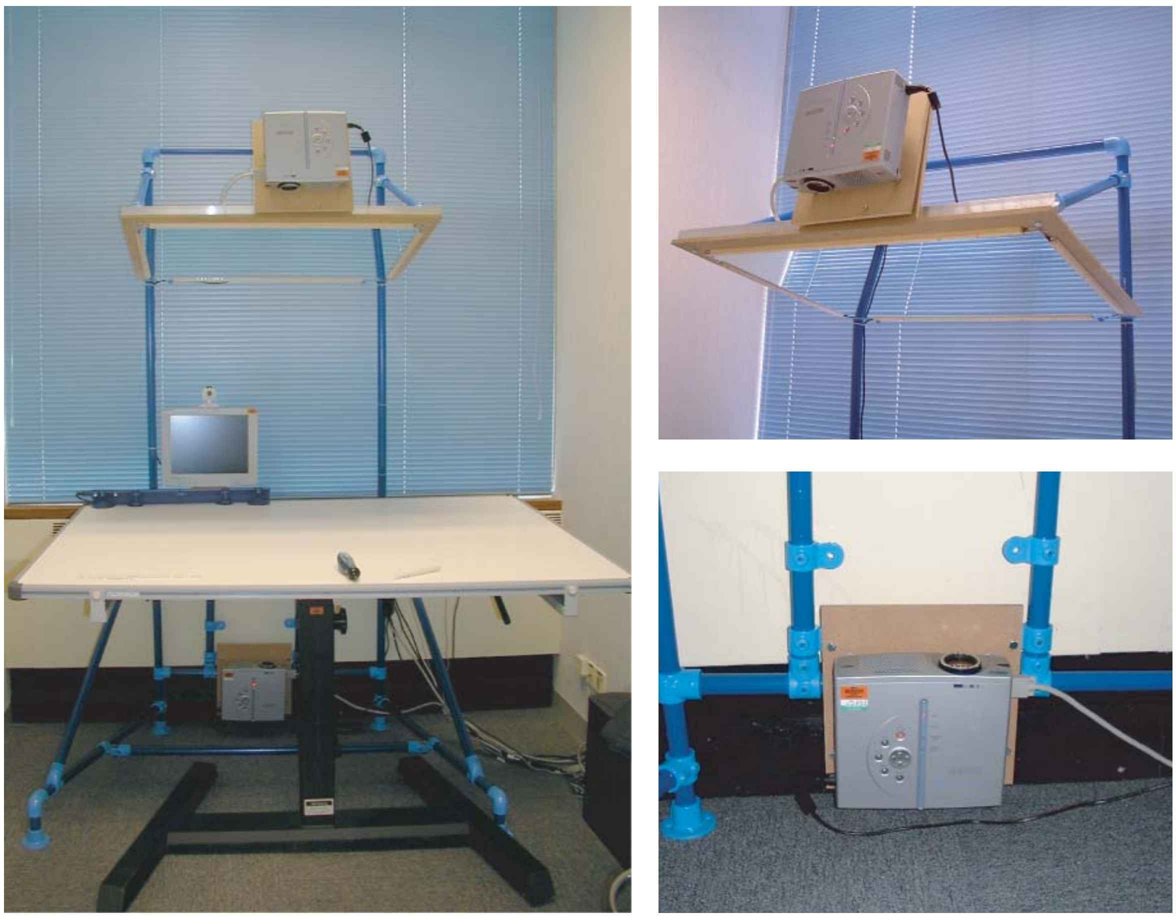


Figure 3.8: The second version of the Escritoire hardware. A purpose-built frame (left) holds the two projectors and the large mirror. The top projector is mounted at a steep angle (top right) so there is no need for a mirror to reflect it's image downwards. The bottom projector (bottom right) points upwards, towards the large mirror.

fovea of A3 size that is 1024 pixels across has a resolution of 62 dpi, so one pixel in the periphery takes the area of approximately eight pixels in the fovea.

Figure 3.7 shows the shelf and top projector of the first version of the Escritoire hardware. Figure 3.8 shows the second version of the hardware. The second version has a purpose-built frame to hold the shelf and projectors, and rather than using a mirror to project down onto the foveal region the top projector is mounted at a sharp angle so it points directly at the desk. Although this violates the instructions for these projectors that state that they should not be placed at an angle of more than 10 degrees (projector details are given in Appendix B), I have not noticed any problem in practice. The minimum throw distance given for the projectors is 1.43m which is too far, even at minimum zoom, to create an image small enough for an A3-sized fovea. However, I have found that a distance of around 0.8 metres is possible—projectors can generally focus on surfaces significantly closer than those for which they have been designed.

Projectors generate significant amounts of heat. The arrangement in Figure 3.8 has two projectors, each of which provides 1100 lumens of light, has a 150 watt bulb, and draws 250 watts in total. It is in a 2.5×5 metre room which was warmed considerably by the projectors, requiring the windows to be opened after about an hour. For long-term use, ducts could be used to channel the hot air from the projectors out of rooms whose air conditioning cannot cope [BW00].

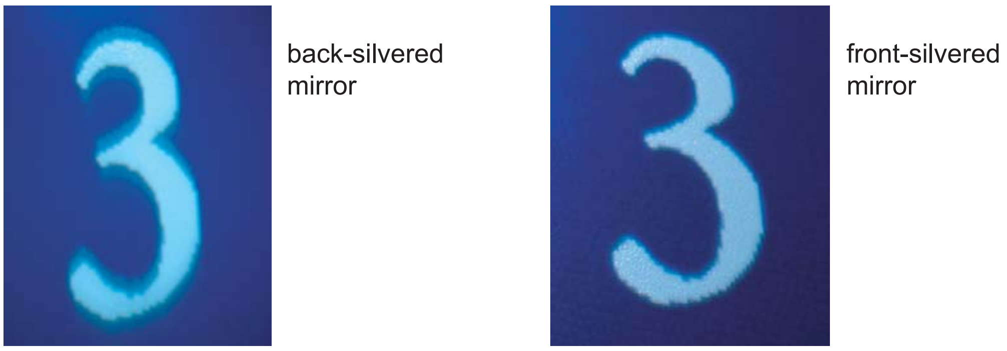


Figure 3.9: Difference in image quality when using front-silvered and back-silvered mirrors. The first image has been reflected onto the desk using a normal, back-silvered mirror and exhibits unwanted multiple reflections. The second has been reflected using a front-silvered mirror and does not have those multiple reflections. Each ‘3’ is approximately 35×60 pixels and its size on the desk is about 30×50 mm.

The small mirror for the fovea projector is front silvered—it has the silver on the outside, rather than protected behind the glass. This type of mirror is more expensive and requires careful handling, but it avoids the multiple reflections that occur when a normal back-silvered mirror is used (Figure 3.9). I have used a back-silvered mirror for the peripheral projector because image quality is not as important, and the much larger size of the mirror would exacerbate the disadvantages of front-silvered mirrors described above. Several other choices allow the quality of the foveal image to be as high as possible while also ensuring that the system is practical to build. The smaller size of the fovea means that its light is spread over a smaller area so it appears brighter. This would be a problem in a multi-projector display wall where the goal is to disguise any difference between the parts of the display provided by different projectors, but for the foveal display the change in intensity coincides with and enhances the change in resolution. The brightness of current projectors and the small size of the display when compared to a wall projection mean that the foveal display can easily be used under normal office lighting, although, as with conventional monitors, direct sunlight should be avoided. When I made the first Escritoire display I had two projectors of different resolutions so I used the higher resolution one for the fovea, which is consistent with the goal of maximizing the quality of the foveal region. Finally, I have configured the graphics warping algorithms for the two projectors to minimize warping artifacts in the fovea: before warping, the graphics are prepared at the resolution of the fovea so artifacts for that projector only occur because of the warp, whereas the graphics for the periphery, which are needed at a lower resolution, are produced by scaling down the high-resolution graphics before warping. The graphics for the fovea are therefore spared the extra processing step. The trade-offs between fovea and periphery are summarized in Table 3.1.

	Fovea	Periphery
display size	small	large
light intensity	high	low
projector resolution	high	low
display resolution	high	low
mirror	front-silvered	back-silvered
graphics scaling	no	yes

Table 3.1: Design trade-offs between fovea and periphery. The quality of the foveal image is paramount. Projector resolution is relevant when the two projectors have different resolutions. Graphics scaling is a step that is avoided for the fovea because it introduces artifacts.

Conventional computer displays evolved from the teletypes of the 1970s, which in turn evolved from typewriters. They are designed to occupy a visual angle of approximately 20 to 40 degrees at the centre of the visual cone and are meant to be read without rotating the neck [SS97]. A much larger display is qualitatively as well as quantitatively different, so simply displaying a larger version of the traditional GUI does not work [PMMH93, page 395]. The Escritoire presents an interface that is modelled on real paper, where there are no controls that are located in distant parts of the display and are therefore unreachable, or permanently outside the fovea and are therefore unreadable. The interface is described in Chapter 6.

3.4 Pen input

To allow items to be manipulated on the Escritoire's desk display, input devices for both of the user's hands are required. Each device should span the entire the desk, and any controller, such as a pen, should not be tethered. Various possibilities for the input devices are given below.

Ultrasonically tracked whiteboard pens are available from eBeam [EB], Mimio [Mim], and Pegasus [Peg]. With these systems, a whiteboard marker pen is placed in a jacket that emits an ultrasound signal when the pen is pressed to the board. Microphones on a bar along the top or side of the board pick up the signal which is used to locate the pen. The systems are designed to record marks that are made on a normal whiteboard, but when combined with a projector they can be used to control a computer. A point to note is that line of sight is required from the pen to the receiver bar. This is not usually a problem for a vertical whiteboard because the only thing that touches the board is the

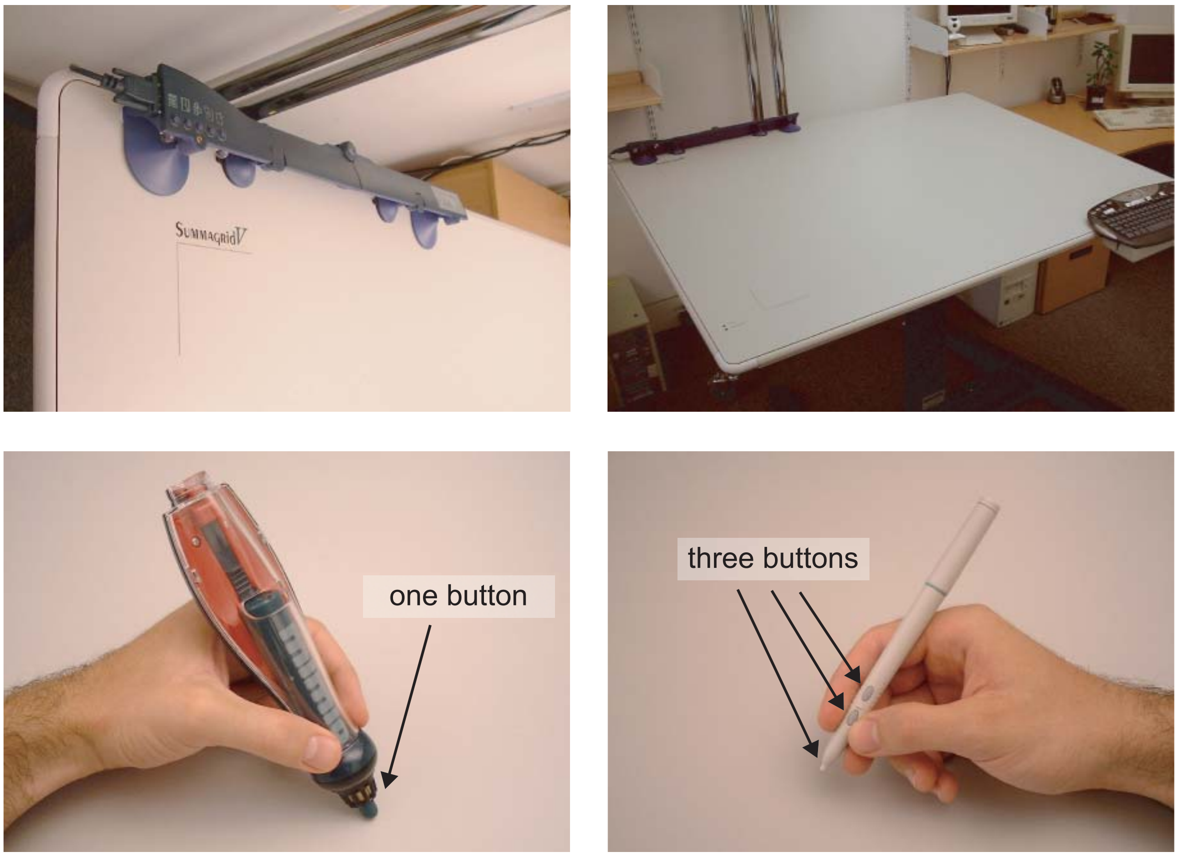


Figure 3.10: Combining a digitizer and an ultrasonic pen to allow two-handed input. The receiver bar for the ultrasonic pen (top left) is fixed along the far edge of the desk. The desk (top right) is a large electromagnetic digitizer whose signals do not interfere with the ultrasonic ones. The ultrasonic pen (bottom left) for the non-dominant hand has a single button to detect when it is pressed to the surface. The digitizer pen (bottom right) for the dominant hand has three buttons to perform different functions.

pen, but with a horizontal desk the user's hands and arms may obstruct the ultrasound signal.

Laser scanning whiteboards are available, such as the Webster LT Series [Web]. A laser sweeps the region just above the whiteboard surface, identifying and locating the pen.

Touch sensitive boards are available which can be used either with a pen, or with the fingers [Web, ST, TB]. They are generally constructed from two sheets of resistive material that are brought together when something touches the board. This technology is not suitable for a desk because the user will often rest his hands and arms on the horizontal surface which would create spurious readings.

The Wacom Intuos tablets are the only commercial devices I know of that allow multiple pens to be used at once. Two can be used simultaneously, but the maximum size of the tablets is only A3. Researchers at Mitsubishi Electric Research Laboratories have created a system called DiamondTouch that allows two users to interact simultaneously with a large surface using their fingers (Section 2.4.4). A user cannot, however, use two hands simultaneously and independently, and they cannot use a pen for precise work like writing.

Digitizers are large surfaces that generally use electromagnetism to track a puck or stylus. GTCO Calcomp [Cal] and Numonics [Num] manufacturer these devices which are used to input data from paper drawings for GIS and CAD applications. They offer high resolutions of up to 500 lines per millimetre, and accuracies of ± 0.25 mm. They are available in A0 size and sometimes even larger.

To create a display for the Escritoire with two pens that can be used simultaneously I have combined an A0 digitizer with a Mimio ultrasonic pen (Figure 3.10). Because the digitizer uses electromagnetism and the Mimio uses ultrasound, the two devices do not interfere with each other. The high-resolution multi-button digitizer pen is for the user's dominant hand which will be the most dexterous, and the lower-resolution single-button ultrasonic pen is for the user's non-dominant hand which should not need such high accuracy. The differences between the two devices are summarized in Table 3.2 below. The large size, coarser resolution, and simple operation of the ultrasonic pen make it suitable for use by the non-dominant hand which Buxton *et al.* [KMB93] have shown to be just as good as the dominant hand for large targets and large distances. The methods for calibrating the devices to the projected displays are described in Chapter 5.

	Dominant	Non-dominant
resolution	high	low
cost	high	low
buttons	3	1
grip of pen	sleek	chunky

Table 3.2: A pen for each of the user's hands. The pen for the dominant hand is used for precise tasks like writing. The pen for the non-dominant hand has lower accuracy and only one button, and is used to move items around on the desk display.

3.5 Summary

I decided to create the display of the Escritoire using projectors because the technology has been advancing steadily in recent years, and portable projectors are now small, light, and cheap enough to be peripherals of personal computers. Initially I made a wall display and created some devices called wands whose position and orientation is tracked in three dimensions using a Polhemus Fastrak. I implemented two methods of mapping the wand position to a point on the display and found that the one in which a line through the wand is intersected with the display surface was the best.

I then created the foveal display by combining two projectors to create an A0-sized display that has an A3-sized high-resolution area for detailed work. Mirrors allow the large projected display to be very compact, and the system is designed produce a good quality image in the foveal region. To achieve simultaneous two-handed input over the entire area of the desk I have combined a large digitizer with an ultrasonic pen. Like the fovea and periphery of the display, the pens for the two hands have intentional differences: the one for the dominant hand has higher resolution, a shape that affords finer control, and multiple buttons.

Chapter 4

Projected displays

As I stated previously, a projector arrangement requiring precise mechanical alignment would be unsatisfactory. Instead, the projected graphics can be prewarped to compensate for the distortion caused by rough alignment of projectors [AR03b]. This chapter introduces the various co-ordinate spaces that must be considered for the various input and output devices, and the mathematical relationship between them. It then describes how a commodity video card with 3D acceleration can be used to apply a projective transform to warp projected graphics. It is critical that the image to be warped can be updated quickly after a change is made to the contents of the desk display, so finally a method is given for improving the frame rate of the Escritoire client by updating the display efficiently.

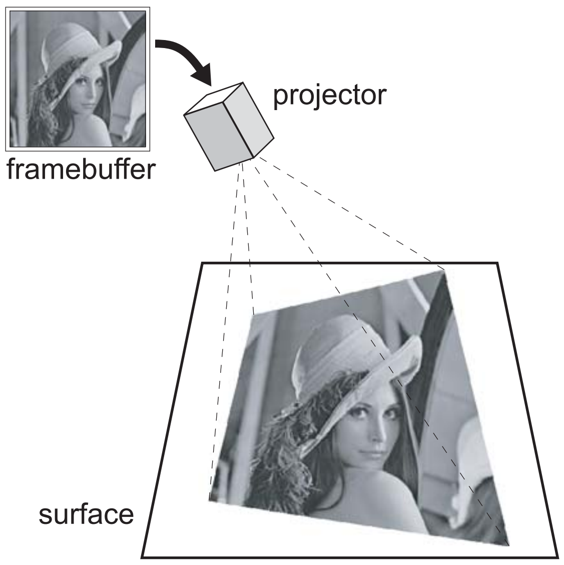


Figure 4.1: The rectangular image from a projector not aligned to the display surface will appear as a distorted quadrilateral.

4.1 Mathematical basis

Projective geometry provides a model for the distortion experienced by an image as it is projected from a digital projector onto a flat surface. The model is formalized below and justified empirically in Section 5.1.4. It uses various two-dimensional co-ordinate spaces, including that of the pen input device used to calibrate the model. The system that is required to map points from one co-ordinate space to another is described below.

4.1.1 Projective geometry

When a projector is used to display a rectangular image on a flat surface, and the projector is aligned so that its axis of projection is perpendicular to the plane of the surface, the image appears rectangular. In oblique projection, where the projector is not aligned in this way, the image appears distorted, as some quadrilateral other than a rectangle (Figure 4.1).

Image distortion

I will assume that the projector exhibits 'pin-hole' optics, making it analogous to a pin-hole camera, but in reverse. The image to be displayed is prepared in the 2-dimensional framebuffer, projected out into the 3-dimensional world, then viewed on the 2-dimensional display surface. Heckbert [Hec89, chapter 2] explains this process in reverse for texture mapping in computer graphics. The projector case is analogous and, as for texture mapping, the transformation from the 2D framebuffer to the 2D display surface can be achieved directly without reference to the intermediate 3D stage.

The projection of an object in the world to the imaging plane of a pin-hole camera, and the beaming of an image from a projector onto an object in the world, are cases of central projection [HZ00, chapter 5], that is, projection with respect to a perspective centre. When a planar surface is captured on a camera's imaging plane, the transformation from one plane to the other has the form,

$$\begin{bmatrix} X' \\ Y' \\ W' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix}, \tag{4.1}$$

where $[x', y', w']^\top$ and $[X', Y', W']^\top$ are the homogeneous representations of the source and destination points respectively. The Cartesian point (x, y) is represented by the homogeneous point (xw', yw', w') for any $w' \neq 0$ so we can use $[x, y, 1]^\top$ as the source point and obtain the destination point as,

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} X'/W' \\ Y'/W' \end{bmatrix}.$$

The 3×3 matrix is called a planar homography and has eight degrees of freedom because its scale is unimportant: multiplying the homogeneous destination point by an arbitrary non-zero scalar has no effect on its 2D location. Sukthankar *et al.* have recently used planar homographies to characterize and correct the distortion experienced by projected images [SSM00, SSM01].

Computing a homography

To compute the homography that represents the distortion experienced by the image projected onto the desk I will use correspondences between points at known locations in the framebuffer and their measured locations on the display surface. The homography matrix could be determined using four point correspondences represented by the pairs

$(x_i, y_i), (X_i, Y_i)$ where $i \in [1, 4]$ and all the w_i are 1, by assuming that $h_{33} = 1$ and noting that,

$$\begin{aligned} X_i &= \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + 1} , \\ Y_i &= \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + 1} . \end{aligned}$$

Four such constraints on the elements of the homography can be combined to form a linear system,

$$\begin{bmatrix} x_0 & y_0 & 1 & 0 & 0 & 0 & -x_0X_0 & -y_0X_0 \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1X_1 & -y_1X_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2X_2 & -y_2X_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3X_3 & -y_3X_3 \\ 0 & 0 & 0 & x_0 & y_0 & 1 & -x_0Y_0 & -y_0Y_0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1Y_1 & -y_1Y_1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2Y_2 & -y_2Y_2 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3Y_3 & -y_3Y_3 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ Y_0 \\ Y_1 \\ Y_2 \\ Y_3 \end{bmatrix} , \tag{4.2}$$

from which h_{11} to h_{32} can be obtained using Gaussian elimination [PTVF92, chapter 2]. The mapping could be determined in this way for an image warping application, or a manual calibration step for a system such as the Escritoire, where the four corners of the warped image are dragged into position using the mouse. Hartley and Zisserman [HZ00, page 71] give a good explanation of projective geometry as it pertains to computer vision, where projective mappings have been used for some time. They also describe a method for using more than four points to calculate a homography. Essentially, Equation (4.2) above is rearranged into the form,

$$\begin{bmatrix} x_0 & y_0 & 1 & 0 & 0 & 0 & -x_0X_0 & -y_0X_0 & -X_0 \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1X_1 & -y_1X_1 & -X_1 \\ & \vdots & & & \vdots & & & \vdots & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_nX_n & -y_nX_n & -X_n \\ 0 & 0 & 0 & x_0 & y_0 & 1 & -x_0Y_0 & -y_0Y_0 & -Y_0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1Y_1 & -y_1Y_1 & -Y_1 \\ & \vdots & & & \vdots & & & \vdots & \\ 0 & 0 & 0 & x_n & y_n & 1 & -x_3Y_3 & -y_3Y_3 & -Y_n \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \mathbf{0} , \tag{4.3}$$

which can be written as,

$$\mathbf{Ah} = \mathbf{0} .$$

The solution $\mathbf{h} = \mathbf{0}$ is not useful, and the scale of the homography is not important, so the constraint $||\mathbf{h}|| = 1$ is used, that is, the magnitude of the vector $\mathbf{h}$ is fixed at 1. If n, the number of point correspondences, is four, there is an exact exact solution for which $\mathbf{Ah} = \mathbf{0}$. If $n > 4$ a least-squares solution for $\mathbf{h}$ is obtained as the eigenvector corresponding to the smallest eigenvalue of $\mathbf{A}^\top\mathbf{A}$. This corresponds to fitting a plane that passes through the origin in $\mathbb{R}^9$ to the nine-dimensional points given by the rows of $\mathbf{A}$, where the value found for $\mathbf{h}$ will be the normal vector to that plane.

Since the magnitudes of the eigenvalues are irrelevant in this case, one can either use the desired eigenvector of $\mathbf{A}^\top\mathbf{A}$, or use Singular Value Decomposition [PTVF92, section 2.6] to convert $\mathbf{A}$ to the form $\mathbf{UDV}^\top$ and use the last column of $\mathbf{V}$. The matrix processed by the first method is guaranteed to be square, symmetric, and of fixed size (9×9), while the second method avoids the matrix multiplication to compute $\mathbf{A}^\top\mathbf{A}$ which is the most time consuming stage of the first method. I normalize the points [HZ00, page 91] in the two 2D spaces before they are used to compute a homography, to avoid unwanted effects due to the choices of co-ordinate frames.

4.1.2 Co-ordinate spaces

The Escritoire processes 2D locations in various co-ordinate spaces that correspond to the various input and output devices, plus one standard space that links them together which is device-independent. Initially I will assume that there is just one projector and one pen input device. The four co-ordinate spaces are shown in Figure 4.2.

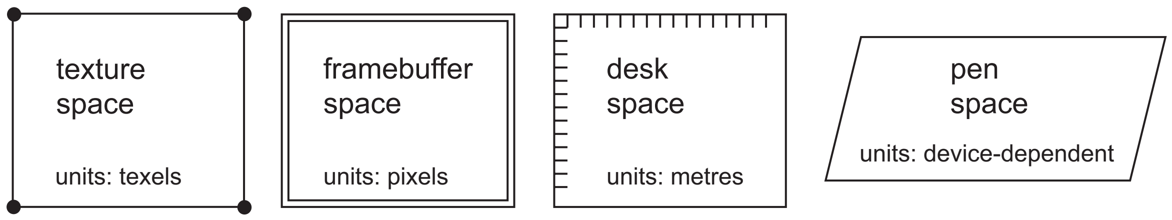


Figure 4.2: The four co-ordinates spaces used by the Escritoire: An image is prepared in texture space then warped to a quadrilateral in framebuffer space; graphical objects are positioned in desk space; input events occur in pen space.

The image to be displayed is created in *texture space*. This is a 2D bitmap in the video card of the computer that is controlling the projector, where the units of measurement are texels. The pixels that are sent to the projector are stored in *framebuffer space*. The image in the texture is warped as it is transferred to the framebuffer in a manner designed to undo the distortion of the final display due to oblique projection. Graphical objects to be displayed on the desk are arranged in *desk space* which is a standard, device-independent co-ordinate space where distances are measured in metres. The size and position of objects in this space, and therefore on the physical display, can be chosen irrespective of the configuration of the input and output devices. Events from the pen input device have locations in *pen space*. This space is where input data for calibration of the projectors are measured, and where new pen events occur which must be transformed to desk space locations to allow interaction with the graphical objects on the desk.

Points must be transformed from one co-ordinate space to another. Figure 4.3 shows the interesting transformations. The projected image starts in the framebuffer and is projected onto the display surface where it will be measured with the pen during calibration, thus the transformation $\mathbf{H}_{\mathrm{fp}}$ from framebuffer to pen space is a planar homography as described in Section 4.1.1. This homography is dictated by the relative positions of the projector and display surface so it cannot be altered. $\mathbf{H}_{\mathrm{fp}}$ is measured during calibration, and the goal of transformation $\mathbf{H}_{\mathrm{tf}}$ from texture to framebuffer is to warp the image, as described below in Section 4.2.2, to undo the distorting effects of oblique projection. The texture holds a rectangular image of the graphical objects that should appear on the desk, and I have defined the origin of desk space to be at the origin of the texture used to display the periphery. This means that for the periphery $\mathbf{H}_{\mathrm{dt}}$ is simply a scaling from metres to texels, which converts the position of a graphical object on the desk to a position on the texture where the corresponding bitmap should be drawn. For the fovea $\mathbf{H}_{\mathrm{dt}}$ also includes a translation. The transformation $\mathbf{H}_{\mathrm{pd}}$

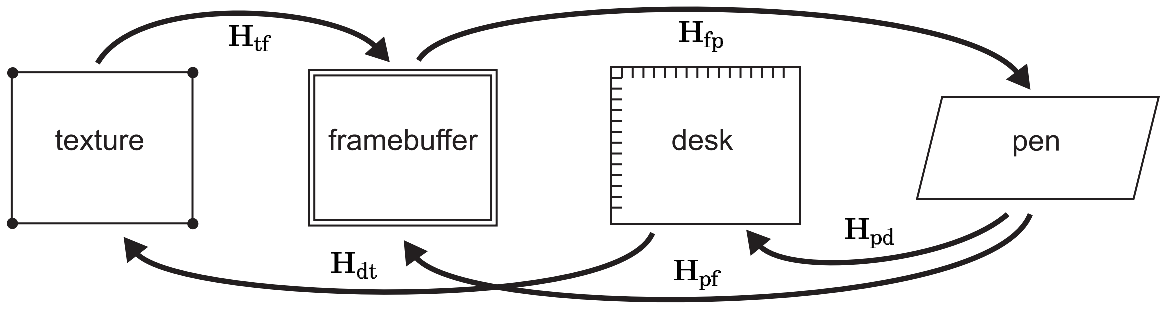


Figure 4.3: Transformations between co-ordinate spaces: the distortion $\mathbf{H}_{fp}$ experienced by the image as it is projected from from framebuffer to pen space is determined by the physical projector set-up, and its inverse $\mathbf{H}_{pf}$ is measured through calibration; the warp $\mathbf{H}_{tf}$ is used to undo the effects of oblique projection; $\mathbf{H}_{dt}$ should be simply a scaling if the distortion in the projected image has been removed; $\mathbf{H}_{pd}$ allows new event locations from the pen to be transformed to points in desk space.

from pen space to desk space is a similarity transformation, that is, a combination of a uniform scaling, a rotation and a translation (see Section 5.1.1 *Global mapping functions* for more information about transformation functions). The scale from device-dependent units to metres is determined from knowledge of the pen input device, and the rotation and translation are determined when the user chooses the position of the rectangular display in pen space, as described below in Section 4.2.1. Since affine and similarity transformations are special cases of projective transformations, all of the transformations in Figure 4.3 can be represented as planar homographies, thus they can be inverted and composed to allow a homogeneous point in any co-ordinate space to be transformed to a corresponding point in any other co-ordinate space.

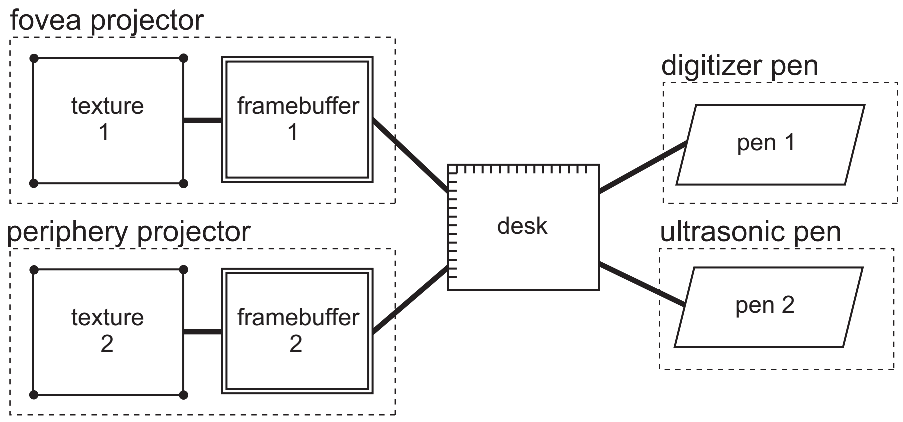


Figure 4.4: Co-ordinate spaces for multiple input and output devices. These are the co-ordinate spaces used for the Escritoire's desk display, which has projectors for the fovea and periphery, and a pen for each of the user's hands. The single desk space where graphical objects are arranged links the system together.

The Escritoire uses multiple projectors and multiple input devices, so the four co-ordinate spaces described above—texture, framebuffer, desk, and pen spaces—actually represent classes of co-ordinate space. Figure 4.4 shows how I have linked the various spaces for a foveal desk display with two projectors and two pen input devices to form a graph that can be traversed by applying the various transformations. The single desk space is reachable from all of the co-ordinate spaces. When a wall display is added more co-ordinate spaces are created that form another graph that has its own separate desk space, and is disjoint from the graph for the horizontal display.

4.2 Implementation

Once the projectors for the Escritoire's foveal display have been roughly aligned, the user is required to provide point correspondences between the display surface and the framebuffer thus calibrating the warp of the projected image, and to choose the position of the image on the display surface. Graphics hardware is then used to apply the warp to the projected images in real time, and various issues are pertinent its use. Finally, an algorithm for efficiently updating the display is necessary because this is the most time-consuming task of the system.

4.2.1 Calibration

Calibrating the foveal display involves two manual stages: selecting projected targets and positioning the display on the surface. A further automatic step blanks the foveal region from the periphery display, so that any point on the display surface is only illuminated by a single projector.

Selecting projected targets

To get the point correspondences necessary to compute the homography from pen space to framebuffer space, a series of targets is projected onto the display surface that the user must select with the pen (Figure 4.5). This process takes approximately 20 seconds for each projector when 9 targets are used for each one. The mapping $\mathbf{H}_{\mathrm{pf}}$ then allows a rectangular image to be projected onto the display surface from an obliquely mounted projector (Figure 4.6).

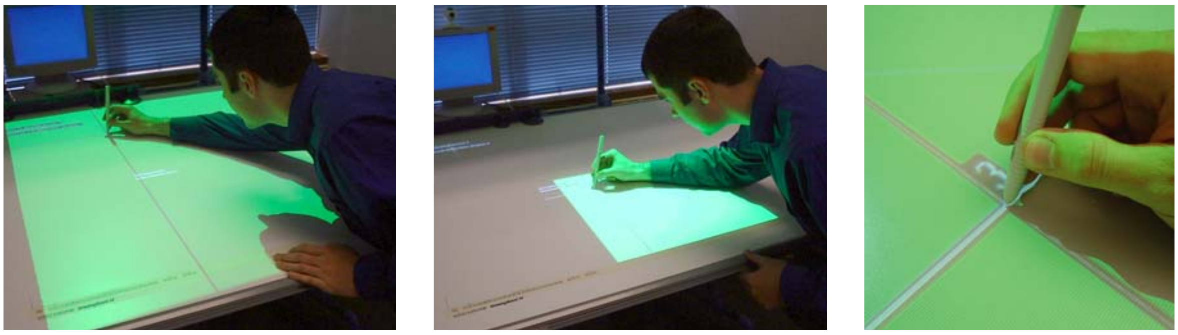


Figure 4.5: The user calibrates the image warping process by selecting targets projected by the (left) peripheral and (centre) foveal projectors. Each target (right) provides one point correspondence that is used in the homography calculation.

Positioning the display

In general, neither the projector's field of view nor the co-ordinate space of the pen input device is aligned with the physical surface. The user must specify a rectangular region of the surface in pen space (for each projector) that will be used as the display (Figure 4.7). In my implementation a reasonable initial position for the display rectangle is derived based on the locations of the calibration points from the previous stage, then the display must be translated and rotated into a suitable position. The translation and rotation parameters, together with the scale to convert from the pen's device-dependent units of distance, are used to create the similarity transformation $\mathbf{H}_{\mathrm{pd}}$ from Figure 4.3.

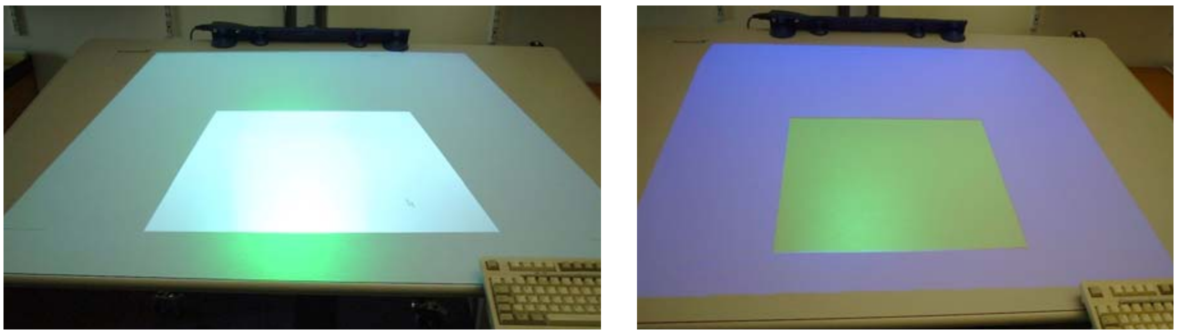


Figure 4.6: Before calibration (left) the output of the projectors is distorted due to oblique projection so the images will not appear rectangular. After calibration (right) the projected images are warped to compensate for oblique projection and to align the display regions with the desk. The foveal region is blanked out in the periphery projector to avoid projecting on that area twice.

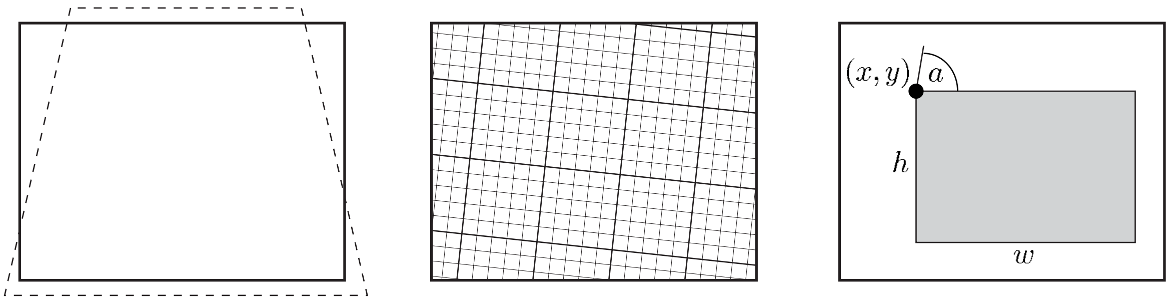


Figure 4.7: Positioning the display rectangle on the surface: (left) the projector's field of view will not coincide with the surface; (centre) in general the pen co-ordinate system will not be aligned to the surface; (right) the user picks the location (x, y), size (h, w), and angle a of the display to make best use of the surface.

The user also chooses the width and height of the display rectangle: these parameters have no effect on $\mathbf{H}_{\text{pd}}$, but they determine the extents of the display.

When transformations $\mathbf{H}_{\text{pf}}$ and $\mathbf{H}_{\text{pd}}$ have been calculated, an image of the graphical objects in desk space is created in the rectangular texture, which is then warped to a quadrilateral in the framebuffer. The corners of that quadrilateral are determined by taking a point $\mathbf{p}$ at each corner of desk space and mapping it to the point $\mathbf{H}_{\text{pf}}\,\mathbf{H}_{\text{pd}}^{-1}\,\mathbf{p}$ in the framebuffer.

Blanking the foveal region in the periphery

The two projectors that form the foveal display overlap in the foveal region. I wish to only use the high resolution, fovea projector in that area, so I blank out that region from the periphery projector. Each time the periphery's texture is warped to produce a quadrilateral in its framebuffer, a black quadrilateral is drawn on top that corresponds to the foveal region. The fact that the co-ordinate spaces and transformations form a connected graph, as shown in Figure 4.3, means that the quadrilateral in the framebuffer of the fovea can simply be transformed to that of the periphery.

4.2.2 Exploiting graphics hardware

When a new event from a pen is received its 2D location is simply mapped to desk space through multiplication by a homography matrix, but when the image prepared in the texture is warped to the framebuffer it must be re-sampled in its entirety. Fortunately, commodity 3D video cards, which are designed for the games market, have hardware support for projective warps because they are necessary for rendering texture-mapped surfaces in 3D scenes.

The general method for using the projective geometry of Section 4.1.1 to warp images, and issues concerning current video cards, are described below. They are followed by examples of, and issues regarding, the two main application programming interfaces that expose 3D video hardware: Direct3D and OpenGL.

General

The description of projective geometry in this document follows the mathematical convention of representing points as column vectors, and pre-multiplying them with transformation matrices. Direct3D and OpenGL both take the opposite approach by using row vectors that are post-multiplied, which is important to remember. This also affects the storage of matrices, which use column-major order rather than the C convention of row-major order. This is apparent in the OpenGL section below.

A projective warp is applied to a rectangular bitmapped image by first preparing it as a texture in the memory of the video card. This is then used to texture map a quadrilateral in the framebuffer which is comprised of two triangles. The width and height of a texture will usually be restricted to powers of two, which is presumably to make MIP mapping simpler. The Escritoire client uses the minimum texture size that is at least as large as the framebuffer, on the assumption that the user will be able to position the projectors and calibrate the display so that most of the field of view of the projector is used. Current video cards have maximum texture dimensions of 512, 1024, or 2048 pixels square which is large enough since the typical resolution for current projectors is 1024×768.

Video cards are designed to have their memory written by the CPU of the computer and read by the digital-to-analogue convertor that sends a signal to the monitor or projector—they are output devices. Reading from the video card is much slower that writing to it, so a rendering algorithm should always cause pixel data to be copied from system memory to video memory and never in the other direction. This precludes, for instance, rendering to an image in video memory then reading it out to system memory to perform extra processing or to transmit it over a network.

When the triangles are drawn in the framebuffer their vertices must be specified. The order of the vertices determines whether the triangle is facing towards or away from the viewer, so they should be ordered so that the triangle faces the viewer, or back-face culling should be disabled, to ensure the triangles are always drawn. Several pieces of information are required for each vertex: the location in the framebuffer, the corresponding homogeneous W value, and the location in the texture to which the vertex corresponds. The location in the framebuffer and homogeneous W value are obtained by mapping the pen-space location of the vertex to framebuffer space using the appropriate projective transformation. The calibration of the Escritoire client's display, described in Section 4.2.1 above, determines the pen-space locations of the vertices. The homogeneous pen-space version of each vertex has the form $(x, y, 1)$, and the corresponding homogeneous vertices (X, Y, W) in the framebuffer are obtained by multiplying by the appropriate homography:

$$\begin{bmatrix} X \\ Y \\ W \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} . \quad (4.4)$$

The vertex in the framebuffer is obtained by dividing by W to get $(X/W, Y/W)$, but W must be retained and associated with the vertex because it will affect the projective warp applied to the texture map. This is because projective texture mapping is usually performed on polygons that are projected from a 3D world to the 2D screen, and the extra homogeneous co-ordinate of the 2D point is dependent on the original z co-ordinate.

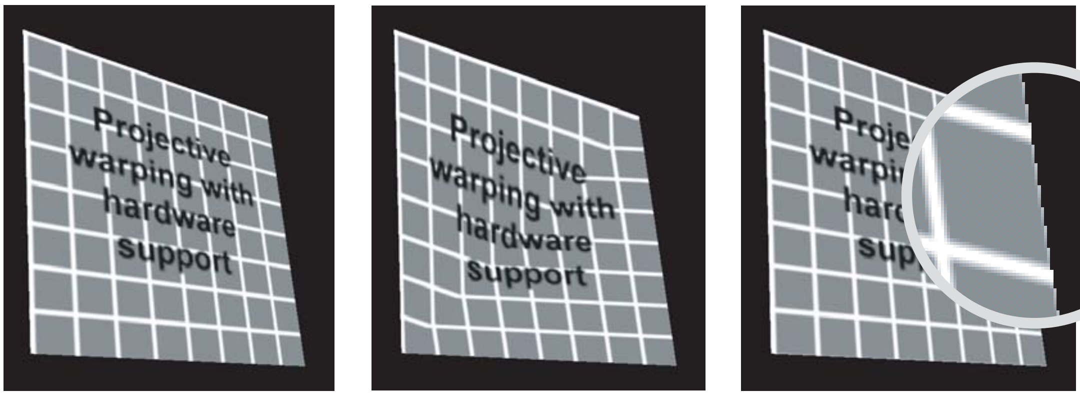

Projective warping with hardware support
Projective warping with hardware support
Proje warpi har sup

Figure 4.8: Errors that can occur when using video hardware for projective warping: (left) the correctly warped image; (centre) the homogeneous W co-ordinates for the vertices have not been used, resulting in inconsistent warps for the two triangles that comprise the image; (right) the texture co-ordinates have been set too high on the right and bottom edges, resulting in the texture beginning to wrap and stray pixels appearing along the those edges.

Researchers on the Tele-Graffiti project have recently used this type of image warping to align a projected image with a sheet of paper on a desk, which is detected using computer vision techniques [TSB02], but they failed to utilize the projective warping now available on video cards, instead dividing the image into 256 quadrilaterals, with each one undergoing an affine warp. Multiple affine warps were used in this manner in early graphics systems [Hec89, page 10] before hardware support was available, but they will not achieve the accuracy of a proper projective mapping. Also, efficiency will be lower and artifacts will occur at the edges of the small quadrilaterals if they are drawn separately due to their ordering, and because the bilinear interpolation used for re-sampling will be broken at the boundaries between the quadrilaterals. Figure 4.8 (left) shows an image that has been correctly warped with a projective transformation. Figure 4.8 (centre) illustrates the error that occurs when the W values for the vertices are not used: the textures of the two triangles have undergone separate affine transformations. Although the vertices of the quadrilateral are in the correct positions, its texture has been warped incorrectly.

Care must still be taken with a single, projectively warped image, that the dimensions of the texture bitmap are within a factor of 2 of its destination dimensions in the framebuffer (Figure 4.9). If the texture is more than twice the size of the quadrilateral on the screen bilinear interpolation will result in pixels in the texture being omitted in the rendered version in the framebuffer. This is a concern for the periphery of the Escritoire because it essentially displays the same graphical display as the fovea but at a lower resolution. Under-sampling in the periphery can be avoided if the periphery texture receives a filtered version of the high-resolution graphical display so that all of the texels equally affect the output in the framebuffer.

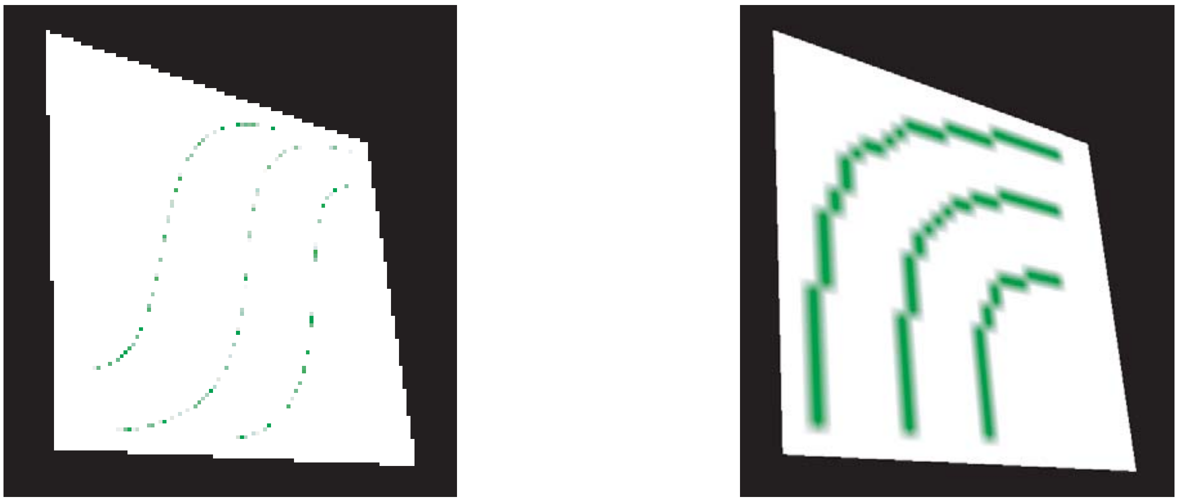


Figure 4.9: The scale factor from texture to framebuffer should be between 0.5 and 2.0. If the texture has too many pixels (left) some of them will be omitted. If the texture has too few pixels (right) the resolution of the projector will not be fully exploited.

In the standard perspective projection used to render 3D scenes the viewer is placed at the origin and looks down the positive z axis, so all visible objects will have positive z values. The W value for each vertex of a projectively warped image is related to its notional z co-ordinate as described above, so it must also be positive. To avoid negative W values one should first note that multiplying a homography by a non-zero scalar does not change the projective transformation it represents: multiplying the 3×3 homography matrix in Equation (4.4) by -1 changes the homogeneous location of a vertex in the framebuffer from (X, Y, W) to $(-X, -Y, -W)$, but both of these correspond to the same non-homogeneous point $(X/W, Y/W)$. During calibration, therefore, I obtain a homography matrix, then multiply it by -1 if it yields negative W values when mapping the calibration points from pen space to framebuffer space.

Once the framebuffer location and the homogeneous W value of a vertex have been determined, just the texture co-ordinates remain to be set. Texture co-ordinates are generally floating-point numbers ranging from 0 to 1 with the origin in the top-left corner of the texture. The top-left vertex of a warped quadrilateral should have the texture co-ordinates $(0, 0)$ and the other three vertices should have co-ordinates $(e, 0)$, $(0, e)$ and (e, e) where e is a number close to one, say 0.999. Textures are designed to be tiled so using $e = 1$ would cause artifacts because stray pixels from the top and left edges of the texture would appear on the bottom and right edges of the destination quadrilateral. Figure 4.8 (right) illustrates the effect.

Direct3D

Microsoft Direct3D [DX] is an application programming interface that controls the 3D graphics acceleration capability of video cards in PCs. The Escritoire client is implemented using this API. It allows the use of the graphics hardware to be optimized for the particular application by exposing various low-level aspects of the process of rendering, such as whether a texture is stored in system or video memory, and whether the application has exclusive control of the video card. It can, however, produce inconsistencies when used with different video cards, and can easily crash the operating system if used in an unintended manner or with erroneous parameters.

Direct3D provides immediate-mode and retained-mode rendering routines. Immediate mode allows simple primitives like lines, triangles, triangle strips and triangle fans to be drawn. Retained-mode allows a scene graph containing many 3D objects to be prepared in advance, then modified and rendered from different viewpoints. This and many other parts of the API are designed to aid the construction of games and other programs that render complex scenes, but for an application like the Escritoire client just the raw speed of the graphics hardware is required, so I have used immediate-mode rendering. A short sequence of calls to create the graphical display from scratch is executed each time the screen is updated.

I store the vertices of the transformed quadrilateral in the framebuffer in C structures with the following definition.

```
struct Vertex
{
   float x, y, z, rhw;     // homogeneous screen co-ordinates
   float tu, tv;           // texture co-ordinates
};
```

The fields `x` and `y` store the 2D position of the vertex in the framebuffer. The depth value `z` is not used, and is always set to 0. The value of `rhw` is the reciprocal of the homogeneous W value obtained from the projective transformation. The texture co-ordinates `tu` and `tv` are set to values between 0 and 1 as described above.

Before the warped quadrilateral is drawn, the shading mode is set to *flat* to avoid using Gouraud or Phong shading in a situation where they would not make sense, and perspective texture mapping is enabled so that the desired projective transformation is applied to every texel of the warped image. The statements that make these changes are shown below, where `p` is a pointer to an object representing the video card.

```
p->SetRenderState( D3DRENDERSTATE_SHADEMODE, D3DSHADE_FLAT );
p->SetRenderState( D3DRENDERSTATE_TEXTUREPERSPECTIVE, TRUE );
```

I use an array of four vertices to create a triangle fan, rather than drawing two separate triangles. One could just as easily use a triangle strip. When the primitive is drawn, a flag is used to specify that the vertices are 'transformed lit vertices' so that no geometric transformations or lighting will be applied to them. The statement used to draw the triangle fan is given below, where, again, `p` is a pointer to the video card object.

```
p->DrawPrimitive(
   D3DPT_TRIANGLEFAN,      // draw a triangle fan
   D3DVT_TLVERTEX,         // use transformed, lit vertices
   v,                      // pointer to array of four vertices
   4,                      // number of vertices present
   0);                     // flag - not used
```

To avoid an error being caused by a vertex being placed off the edge of the framebuffer during calibration, a clipper object is associated with the framebuffer during initialization of the graphics device. The periphery display is created by drawing the appropriate texture as a quadrilateral, then drawing another quadrilateral to mask out the fovea region. The second quadrilateral is drawn without using a texture so it is simply filled with black pixels.

OpenGL

OpenGL [WND99] is a cross-platform standard for 3D rendering. Like the immediate mode part of Direct3D, it provides low-level primitives that can be combined to create complex scenes, although for projectively warping images only the rendering of a few simple primitives is needed.

The warp applied to the texture, $\mathbf{H}_{\mathrm{tf}}$ in Figure 4.3 on page 61, can be put on the OpenGL matrix stack as follows, where `h` is a $3{\times}3$ matrix in row-major order representing the planar homography $\mathbf{H}_{\mathrm{tf}}$, and `g` is a $4{\times}4$ matrix in column-major order representing a transformation of a similar type on 3D homogeneous points.

```
GLdouble g[16];
g[0]=h[0],  g[4]=h[1],  g[ 8]=0,  g[12]=h[2],
g[1]=h[3],  g[5]=h[4],  g[ 9]=0,  g[13]=h[5],
g[2]=0,     g[6]=0,     g[10]=0,  g[14]=0,
g[3]=h[6],  g[7]=h[7],  g[11]=0,  g[15]=h[8];
glMultMatrixd( h );
```

In general, OpenGL's $4{\times}4$ matrices specify transformations on 3D points but in this case the null column and null row show that the z co-ordinate is not being used. Before drawing the image, texturing must be enabled, and the actual texture to be warped must be selected, as shown below. The texture is generated during initialization, which produces an integer identifier `t` that is used to activate the texture before each instance of the quadrilateral is drawn.

```
glEnable( GL_TEXTURE_2D );
glBindTexture( GL_TEXTURE_2D, t );   // t identifies the texture
```

To draw the quadrilateral its vertices are positioned as if it is being drawn directly over the image in the texture, then the $4{\times}4$ matrix performs the transformation from texture to framebuffer co-ordinates.

OpenGL has a specific primitive for drawing quadrilaterals, specified using the constant `GL_QUADS`, which is demonstrated below. Each vertex's 2D location is preceded by it's corresponding texture co-ordinates. The values `W` and `H` are the width and height of the texture, and `E` is the extent of the texture used in each direction. `E` is set to a number slightly lower than one to avoid artifacts due to the texture wrapping as described above.

```
glBegin(GL_QUADS);
  glTexCoord2f( 0, 0 );    // vertex 1
  glVertex2f  ( 0, 0 );
  glTexCoord2f( E, 0 );    // vertex 2
  glVertex2f  ( W, 0 );
  glTexCoord2f( E, E );    // vertex 3
  glVertex2f  ( W, H );
  glTexCoord2f( 0, E );    // vertex 4
  glVertex2f  ( 0, H );
glEnd();
```

4.2.3 Display updates

I have implemented the Escritoire client on a single PC with multiple video cards—see Appendix B for the precise hardware used. Due to restrictions in DirectX I originally used one AGP card and one PCI card, but it is now possible to use one of the dual-head AGP cards that are available. Commodity video cards can apply projective warps to images very quickly, and the speed of the warp is not affected by the use of the slower PCI bus because the texture mapping process is local to the video card. My original hardware configuration takes between 0.15 and 0.45 milliseconds to produce a warped image that fills most of a 1024×768 framebuffer and thus contains around 750,000 pixels. The critical stage of rendering is not performing the texture mapping but *updating* the texture. The system takes 1 to 3 ms to update the texture on the AGP card after a 640×480 image on the desk has moved, and 3 to 10 ms to update the texture on a PCI card. This allows 30 frames per second to be achieved in practice, which includes the whole rendering procedure for two video cards plus the overhead of the rest of the system.

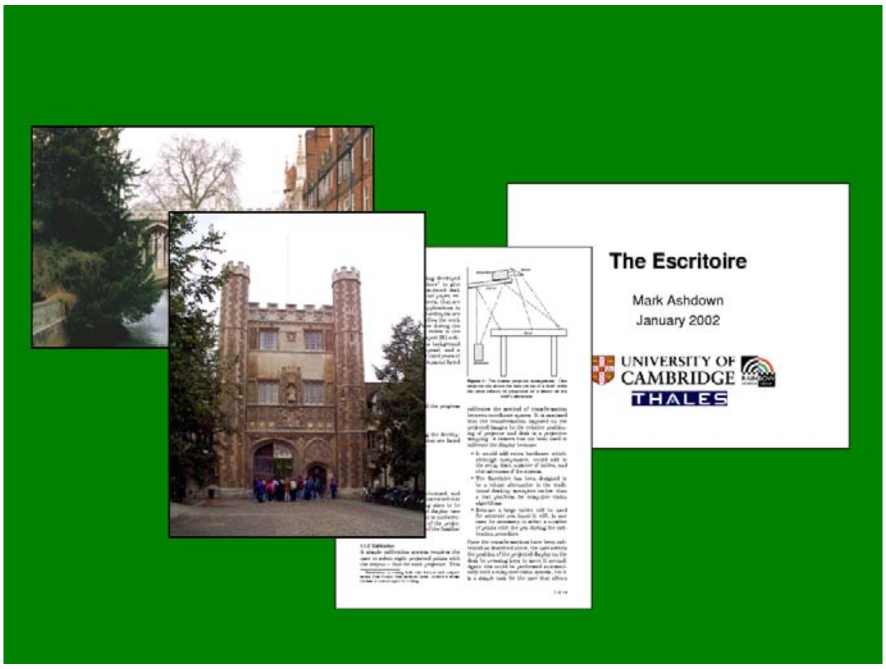


Figure 4.10: The image to be warped by the 3D video hardware contains overlapping bitmaps that represent the sheets of virtual paper. This image in texture memory must be updated quickly to achieve a high frame rate.

The texture contains the image to be displayed on the desk, which is a collection of rectangular bitmaps that represent the sheets of virtual paper which can be placed on top of each other causing occlusion (Figure 4.10). The simplest method to update the texture is to draw each sheet in turn, from bottom to top, so that the one on the top is drawn last and therefore cannot be occluded. That method is described briefly below, and is followed with a new, better method that I use to achieve the texture update times given above.

Bottom-up method

The simplest way to create an image in the texture containing all of the sheets with the required overlaps is to draw them from the bottom, up, using the *painter's algorithm*. If there are n sheets, with an average of p pixels each, this will take time proportional to np, that is, time $O(np)$. Since the texture need only be updated after a change occurs the method can be improved by using an update rectangle: if part of a sheet changes its appearance the update rectangle becomes a region of the texture that encloses the affected area; if a sheet moves, the update rectangle is set to encompass the old and new locations of the sheet; and if multiple changes occur before the display is refreshed, each newly computed update rectangle is merged with the old one by taking the smallest rectangle that encompasses both the old and new areas. When the sheets are drawn they are clipped to the update rectangle first, thus only the pixels in the affected area are redrawn. Each pixel, however, may still be written up to n times with the bottom-up method.

Top-down method

The top-down method described here ensures that each pixel in the texture is written only once, so the number of pixel writes is independent of the number of sheets n. It is similar to a method used in a Window Manager that was created to support real-time applications [Ber93] where multiple layers for the screen are stored, with each real-time application's window having a layer to itself. In that system the layers were combined by specialized video hardware but my system does this in software. The algorithm

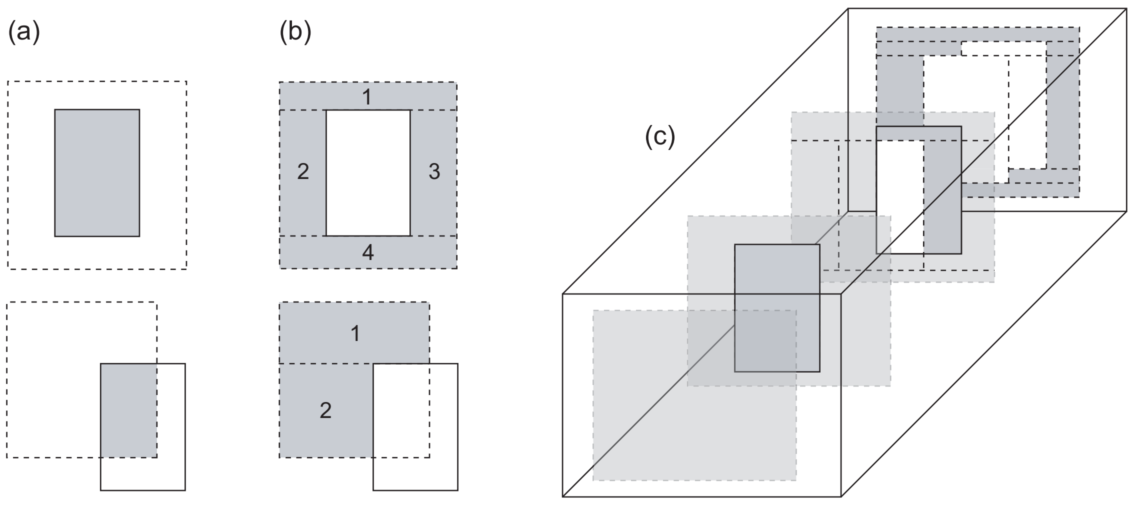


Figure 4.11: Top-down texture updates: (a) a sheet is intersected with an update rectangle to get the area that should be filled; (b) up to four residual rectangles remain; (c) residuals from each sheet are passed to those behind it until the entire original update rectangle is filled.

processes the sheets of virtual paper in order from the top, down. For each one it draws the visible part of the sheet that is contained within the original update rectangle as a number of rectangular parts, before moving to the next sheet. The sheet on the top need only be clipped to the original update rectangle before being drawn, by taking the intersection of the update rectangle and the bounding rectangle of the sheet (Figure 4.11(a)). This intersection will produce up to four residual areas that have yet to be filled (Figure 4.11(b)). The residual areas are kept in a list and passed to the next sheet, the drawing of which may produce further residual areas which are passed on (Figure 4.11(c)). The algorithm terminates when there are no more areas to fill, or there are no more sheets left in which case the remaining areas are filled with the background colour or image. Pseudo-code for this top-down algorithm is given below.

```
make two lists for storing rectangles, called *current* and *pending*
put original update rectangle in *current*
S = the top sheet
**while** *current* is not empty **do**
  take a rectangle from *current*
  intersect the rectangle with S
  if the intersection area is not empty, draw the appropriate part of S
  add up to four residual rectangular areas from the intersection to *pending*
  **if** *current* is empty and there are more sheets to process **then**
    S = the next sheet
    move the contents of *pending* to *current*
  **end if**
**end while**
```

The update regions generated in this way are analogous to the expose events generated by an X server when a window moves or is brought to the top or bottom, although they are not sent between client and server as the expose events are, but are just used within the server. The contents of a tile are retained on the server even when it is occluded, so it is like an X window whose *backing store* attribute is set to *always* [Man93, pages 302–303]. The update regions that are generated cover exactly the area that needs to be redrawn for each tile, rather than a superset of that area, an effect that is achieved in X by setting the *compress_exposure* field to *false* so multiple expose events for a window are not replaced by their union [NO92, pages 225–226].

Optimizations

The rectangles created by the top-down algorithm can be represented as a tree, where each node is a rectangle, and has up to four children representing the residual areas left after the appropriate sheet has been used to fill as much of the rectangle as possible. The root of the tree is the original update rectangle and a rectangle's children are all contained within its bounds. The tree has a branching factor of four in the worst case, and a depth proportional to the number of sheets, n.

Because I use a uniform colour for the background of the desk display, I save time by first drawing the background over the entire original update rectangle, which is a fast operation. The top-down algorithm will then terminate when there are no more sheets left to draw: there may be many small rectangles left that correspond to the exposed pieces of the background, but they can be discarded because the background has already been drawn.

The method described here performs a breadth-first traversal of the tree of rectangles. The list called *pending* that stores rectangles that are due to be processed can, in the worst case, take space $O(4^n)$. While this has not been a problem in practice, the algorithm could be changed to a recursive form that traverses the tree in a depth-first manner, which would have the same time complexity but only take space $O(n)$.

4.3 Summary

A projector can be considered the opposite of a pin-hole camera, and a 2D image projected onto a flat surface is transformed by a planar homography during the transfer from framebuffer to surface. I have described how a homography can be computed from four or more point correspondences, the 2D co-ordinate spaces that should be considered for a projected display, how these can be linked to form a graph when multiple projectors and input devices are used, and the stages I have used to calibrate the Escritoire's display so the images from the two projectors are corrected for distortions due to oblique projection and so they are aligned with one another. I have described implementation details for the graphics warping such as the use of column vectors, choosing the size of the textures on the video card, writing to video memory, using correct homogeneous co-ordinates, and using correct texture co-ordinates. I have also described issues specific to Direct3D—the use of immediate-mode rendering, storage in C structures, setting up the video card, and drawing with triangle fans—and issues specific to OpenGL—placing a planar homography on the 4×4 matrix stack, using textures, and drawing quadrilaterals. Finally, because the time taken to update the texture is critical, I have explained the basic bottom-up *painter's algorithm* and have then given a top-down method that is much more efficient.

Chapter 5

Pen input

The user of the Escritoire is provided with 2D and 3D pen input devices, but before they can be used to manipulate the graphical objects on the desk and wall displays the devices must be calibrated. This chapter describes the methods I have used to perform the calibration and to map newly received pen data to 2D events.

5.1 2D input

Each display on the Escritoire has one *primary* input device and any number of *secondary* input devices. The primary input device determines the manner in which the projected graphics are warped, so should it should be the most accurate device for the display. For the desk display described in Sections 3.3 and 3.4 the digitizer is the primary input device. The goal of the calibration is to determine a 2D to 2D mapping from pen space—the plane in which pen events are generated—to framebuffer space—the planar image that is sent to the projector. The projected graphics for the display are then warped in the framebuffer so that they appear correctly in pen space. Figure 4.3, page 61, depicts the relationship between the co-ordinate spaces. Since the digitizer provides accurate readings on a regular grid, the projected graphics that are warped to coincide with this grid will look correct when viewed by the user. The location of the projected display is defined in pen space. The bitmap that is to be displayed is warped as it is prepared in the framebuffer, and the mapping used to do this must be supported by the video hardware. I have used a projective mapping (Section 4.2.2) that is calibrated by measuring the mapping from pen space to framebuffer space by selecting a series of points (Section 4.2.1). The secondary input devices for a display are also calibrated by obtaining a mapping from pen space to framebuffer space, but because this is only used to transform points generated from input device events there is no restriction on the form of the mapping. The ultrasonic pen is a secondary device on the desk display. I have found that the co-ordinates it produces exhibit systematic errors so I have explored several mapping functions. This section describes several 2D mapping functions and the results of using them to calibrate the digitizer and the ultrasonic pen to a projector.

5.1.1 Global mapping functions

A pen input device could be calibrated with a similarity transformation. This would map pen-space points from the pen (x, y), to framebuffer points (X, Y), using the equation

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix},$$

where there are four degrees of freedom that produce a uniform scaling of s, a rotation of θ and a translation of (t_x, t_y). Three-element vectors are used to represent the points in homogeneous form, although the homogeneous (bottom) value is always equal to one in this case. An affine transformation is more general than a similarity transformation, and yields an equation of the form,

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix},$$

where there are now six degrees of freedom. An affine transformation also allows non-uniform scaling and shears. Figure 5.1 shows examples of similarity and affine transformations. For a display that uses oblique projection, where the projector is not pointing perpendicularly at the surface, similarity or affine transformations will not be able to capture the warping experienced by the image. Projective transformations can capture such a mapping and are described below.

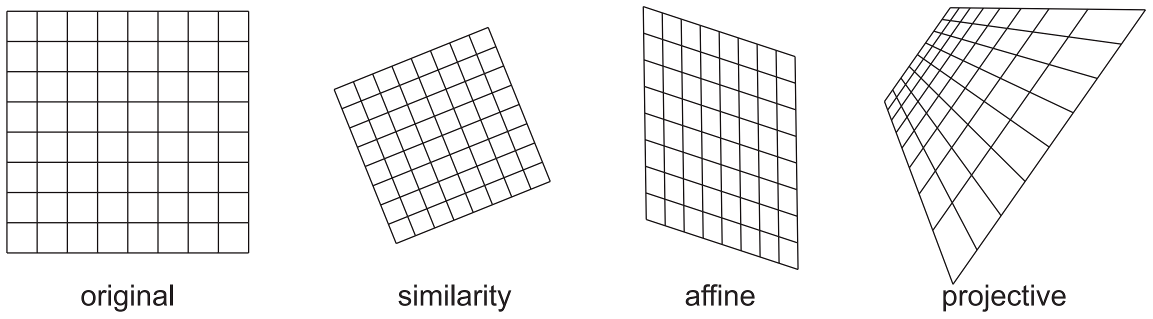


Figure 5.1: Three global mapping functions that could be used to calibrate a pen input device. A similarity transformation scales, rotates and translates the grid, an affine transformation also performs non-uniform scaling and shears, and a projective transformation adds perspective effects.

Projective mapping

A projective mapping from points (x, y) to points (X, Y) has the form,

$$\begin{bmatrix} X' \\ Y' \\ W' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix},$$

from which X and Y can be obtained:

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} X'/W' \\ Y'/W' \end{bmatrix}.$$

All nine of the elements of the 3×3 matrix can be set, including those in the bottom row that determine the perspective effects. The matrix is a planar homography like that used to calibrate the mapping from the primary pen input device on the Escritoire's desk display, that is the digitizer, to the framebuffer, as described in Section 4.2.1. The matrix has nine elements but only 8 degrees of freedom, because it can be scaled without changing the transformation. The digitizer produces location data on a regular grid with an accuracy of ±0.25mm. I calibrate it by clicking on a number of projected targets to provide pairs of pen-space points (x, y) and corresponding framebuffer points (X, Y). I have assumed that the warping experienced by the projected image is a planar homography (Section 4.1.1), and have validated this assumption in Section 5.1.4. The homography is used to warp the projected graphics so that they look correct to the user, and is also used to map new events from the digitizer to corresponding locations in the framebuffer. Once the framebuffer location of a new pen event is known, the location in desk-space can be obtained through the use of homographies between other co-ordinate spaces as shown in Section 4.1.2.

The digitizer provides accurate readings so this method works well for calibrating it, but the readings from the ultrasonic pen include systematic error, that is, an error function that does not change over time. A different mapping function is needed that can account for both the perspective distortion of the projected image and the error in the co-ordinates reported by the pen.

Bivariate polynomial mapping

As explained at the start of Section 5.1 above, because the ultrasonic pen is a secondary input device its mapping function is not restricted to a particular form. The mapping that Wellner used for the DigitalDesk [Wel94, Wel93b] can be written as,

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 \\ \alpha_5 & \alpha_6 & \alpha_7 & \alpha_8 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ y \\ xy \end{bmatrix} .$$

The eight constants α_1 to α_8 define the mapping. Wellner used four point correspondences to get a set of simultaneous equations that was solved with Gaussian Elimination [PTVF92, PTVF02], but a least-squares solution from four or more correspondences can be obtained using a method similar to that in Section 4.1.1 for calculating planar homographies: a single point correspondence provides a linear constraint on the constants, and a set of four point correspondences $\{[(x_i, y_i), (X_i, Y_i)] : 0 \leq i \leq 3\}$ can be stacked to form a 8×8 matrix where

$$\begin{bmatrix} 1 & x_0 & y_0 & x_0y_0 & 0 & 0 & 0 & 0 \\ 1 & x_1 & y_1 & x_1y_1 & 0 & 0 & 0 & 0 \\ 1 & x_2 & y_2 & x_2y_2 & 0 & 0 & 0 & 0 \\ 1 & x_3 & y_3 & x_3y_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & x_0 & y_0 & x_0y_0 \\ 0 & 0 & 0 & 0 & 1 & x_1 & y_1 & x_1y_1 \\ 0 & 0 & 0 & 0 & 1 & x_2 & y_2 & x_2y_2 \\ 0 & 0 & 0 & 0 & 1 & x_3 & y_3 & x_3y_3 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \\ \alpha_8 \end{bmatrix} = \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ Y_0 \\ Y_1 \\ Y_2 \\ Y_3 \end{bmatrix} .$$

This could be solved using Gaussian Elimination but, by modifying the equation to get

$$\begin{bmatrix} 1 & x_0 & y_0 & x_0y_0 & 0 & 0 & 0 & 0 & -X_0 \\ 1 & x_1 & y_1 & x_1y_1 & 0 & 0 & 0 & 0 & -X_1 \\ 1 & x_2 & y_2 & x_2y_2 & 0 & 0 & 0 & 0 & -X_2 \\ 1 & x_3 & y_3 & x_3y_3 & 0 & 0 & 0 & 0 & -X_3 \\ 0 & 0 & 0 & 0 & 1 & x_0 & y_0 & x_0y_0 & -Y_0 \\ 0 & 0 & 0 & 0 & 1 & x_1 & y_1 & x_1y_1 & -Y_1 \\ 0 & 0 & 0 & 0 & 1 & x_2 & y_2 & x_2y_2 & -Y_2 \\ 0 & 0 & 0 & 0 & 1 & x_3 & y_3 & x_3y_3 & -Y_3 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \\ \alpha_8 \\ 1 \end{bmatrix} = \mathbf{0} \, ,$$

a null-space problem is created that has a closed-form least-squares solution calculated in the same manner as the projective mapping. More than four point correspondences can be stacked to form the $2n \times 9$ matrix on the left, which I will call $\mathbf{A}$, so all of the point correspondences that are available can contribute to the result. The least-squares solution is the eigenvector corresponding to the smallest eigenvalue of $\mathbf{A}^\top\mathbf{A}$. The nine-element solution vector is scaled so that its bottom element is equal to one, then the values of α_1 to α_8 are read from the other eight positions (any multiple of the solution vector will also minimize the sum of squares). In this case the value being minimized is

$$\sum_i (\alpha_1 + \alpha_2 x_i + \alpha_3 y_i + \alpha_4 x_i y_i - X_i)^2 + (\alpha_5 + \alpha_6 x_i + \alpha_7 y_i + \alpha_8 x_i y_i - Y_i)^2 \, ,$$

which is the sum of squared distances between desired and actual locations for the mapping—the true geometric error for the set of point correspondences, unlike the projective case where the value that is minimized is an algebraic error. The mapping above that uses bivariate polynomials with four terms, which I will refer to as BP4, is a specialization of a bivariate second-order polynomial mapping (BP6) that can be expressed as,

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} \beta_1 & \beta_2 & \beta_3 & \beta_4 & \beta_5 & \beta_6 \\ \beta_7 & \beta_8 & \beta_9 & \beta_{10} & \beta_{11} & \beta_{12} \end{bmatrix} \begin{bmatrix} 1 \\ x \\ y \\ xy \\ x^2 \\ y^2 \end{bmatrix} ,$$

where the twelve constants β_1 to β_{12} can be obtained in the same manner using at least six point correspondences. A global mapping function of this type is attractive because a least-squares solution is readily obtained using linear algebra, but it does require assumptions about the form of the function by which the pen co-ordinates are perturbed. The following section describes a local mapping function that does not require such assumptions.

5.1.2 Local mapping functions

When looking for a mapping from pen space to framebuffer space one can think of pen space as a distorted version of the Euclidian framebuffer space where the axes are perpendicular and points are distributed on a regular grid. The task is to remove the distortion. This is the goal of image registration which is used as a step in many image processing systems to register an image to a standard co-ordinate frame, or to register two images with each other [Bro92]. In many computer vision systems, retrieving the correspondence between feature points in two images is important because this data is not initially available, but the calibration of a display such as that of the Escritoire does not require this step because the correspondence between each pen event and the projected target that was selected is known from the beginning.

The methods described in Section 5.1.1 above map all points with a single transformation. That transformation is generated from a single computation using all of the point correspondences equally. Local mapping functions are different because they split the plane into pieces and apply a different transformation to each piece. They do not make such strong assumptions about the form of the distortion, and can therefore handle more types of distortion.

Possibly the simplest local mapping scheme is one in which, following the arrival of a new pen event, the nearest pen-space control point to the event is selected, and the location of the event is transformed with the same translation as would be required to move the control point to its corresponding framebuffer position. Essentially, the Voronoi diagram of the control points would partition pen-space into regions to which the transformation functions of the form,

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix},$$

would be applied (Figure 5.2), where t_x and t_y represent the translation. This would be a poor mapping function because when the pen moved from one Voronoi region to another there would be a discontinuous jump in the transformed location of the pen events, causing an effect like the *popping artifacts* experienced with some computer graphics systems when, for instance, an object visibly switches between representations at different levels of detail. The mapping for the pen would not be continuous, but below I describe a piecewise linear interpolation scheme that provides a local mapping function that is continuous.

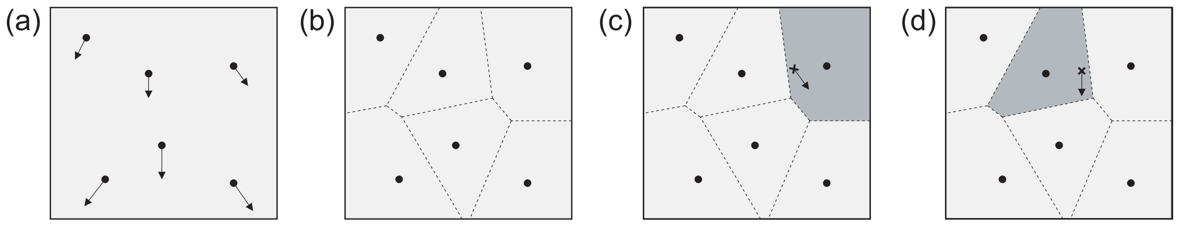


Figure 5.2: A simple local mapping function that maps a new point using the translation associated with the nearest control point: (a) some control points are obtained together with the translations necessary to move them to their desired points in the framebuffer; (b) The Voronoi diagram of the control points partitions the plane in which new events occur; (c),(d) a new event is mapped using the translation for its nearest control point.

5.1.3 Piecewise linear interpolation

The mapping scheme used to undo the distortions experienced by points from the ultrasonic pen of the Escritoire is called piecewise linear interpolation. The scheme, referred to here as PLI, is based on one by Ardeshir Goshtasby [Gos86] with modifications to handle points outside the convex hull of the control points. It has modest computational requirements because the number of control points is limited by the time the user is willing to spend calibrating the system. The scarce resource when calibrating the pen input device is the user's time, so a scheme that requires the input of a dense grid of points is undesirable. PLI allows the user to start by entering only four points, and to then add extra points as necessary at the positions where the error is perceived to be greatest. It is thus an adaptive scheme where the user decides when the mapping is good enough, and where new points can be added to improve the mapping whenever the user notices that an improvement is required.

There are three steps for initializing the mapping:

- Triangulate the control points in pen space to create a set of regions inside the convex hull of the points.
- Determine the mapping function for each inner region in pen space that will map it to its destination in the framebuffer.
- Create unbounded outer regions from the triangles on the convex hull, which will allow the entire pen space to be covered.

There are two steps for mapping a new pen event:

- Determine which inner or outer region the event is in.
- Map the event to a corresponding framebuffer point.

The various steps are described below.

Triangulating the control points

The triangular inner regions inside the convex hull of the calibration points are created with a Delaunay triangulation which is the dual of the Voronoi diagram[1]. Although algorithms are available to compute the triangulation in $O(n\ log\ n)$ time, such as that described by O'Rourke [O'R94], because the points used in this application will number around 20 at most, I have used a a simple, brute force, $O(n^4)$ algorithm. The algorithm comprises about 30 lines of Java code and causes no perceivable delay in the calibration process. The process is illustrated in Figure 5.3. The Delaunay triangulation maximizes the minimum angle over all of the triangles [O'R94, section 5.5.2] thus avoiding very thin triangles as much as possible, and has other desirable properties [HL93, section 9.3.1].

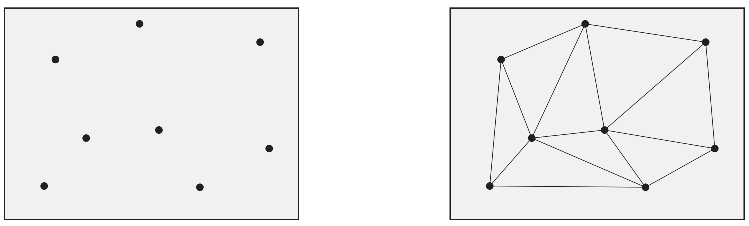


Figure 5.3: Delaunay triangulation: The control points (left) are joined to form a triangular mesh (right). This allows new events inside the convex hull of the control points to be assigned to an inner region, after which they are mapped using linear interpolation.

[1]Goshtasby's paper specifies the use of Theissen regions and Dirichlet tessellation: these are equivalent to the Delaunay triangulation.

Points inside the convex hull

Each new pen event at a location (x, y) must be mapped to a point (X, Y) in the framebuffer. First the inner region in which the event falls is found, then the event is mapped using a linear function that combines the previously measured locations of the three vertices of the region. The three vertices provide point correspondences: $[(x_1, y_1), (X_1, Y_1)]$, $[(x_2, y_2), (X_2, Y_2)]$, $[(x_3, y_3), (X_3, Y_3)]$. A plane that passes through the points (x_1, y_1, X_1), (x_2, y_2, X_2), (x_3, y_3, X_3) is used to map the x co-ordinates, and a similar one is used for the y co-ordinates, giving the equations,

$$\begin{aligned} ax + by + cX + d &= 0\,, \\ ex + fy + gY + h &= 0\,, \end{aligned} \tag{5.1}$$

where the constants a to h can be calculated as the determinants of the following matrices,

$$a = \begin{vmatrix} y_1 & X_1 & 1 \\ y_2 & X_2 & 1 \\ y_3 & X_3 & 1 \end{vmatrix}, \quad b = -\begin{vmatrix} x_1 & X_1 & 1 \\ x_2 & X_2 & 1 \\ x_3 & X_3 & 1 \end{vmatrix}, \quad c = \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}, \quad d = -\begin{vmatrix} x_1 & y_1 & X_1 \\ x_2 & y_2 & X_2 \\ x_3 & y_3 & X_3 \end{vmatrix},$$

$$e = \begin{vmatrix} y_1 & Y_1 & 1 \\ y_2 & Y_2 & 1 \\ y_3 & Y_3 & 1 \end{vmatrix}, \quad f = -\begin{vmatrix} x_1 & Y_1 & 1 \\ x_2 & Y_2 & 1 \\ x_3 & Y_3 & 1 \end{vmatrix}, \quad g = \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}, \quad h = -\begin{vmatrix} x_1 & y_1 & Y_1 \\ x_2 & y_2 & Y_2 \\ x_3 & y_3 & Y_3 \end{vmatrix}.$$

Since $c = g$ the mapping from pen point (x, y) to framebuffer point (X, Y) can be written as,

$$\begin{bmatrix} X \\ Y \end{bmatrix} = -\frac{1}{c} \begin{bmatrix} a & b & d \\ e & f & h \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

Points outside the convex hull

Goshtasby's paper does not adequately describe the details of mapping points outside the convex hull, so I propose a procedure here. The triangulation of the calibration points covers all locations within their convex hull. To map points outside the convex hull we note that each edge on the hull belongs to a triangle whose mapping planes can be extended outwards to infinity to create an outer region. The planes from neighbouring triangles intersect at straight lines. For a continuous mapping—one where there are no holes or overlapping areas in the resultant framebuffer space—a point on one

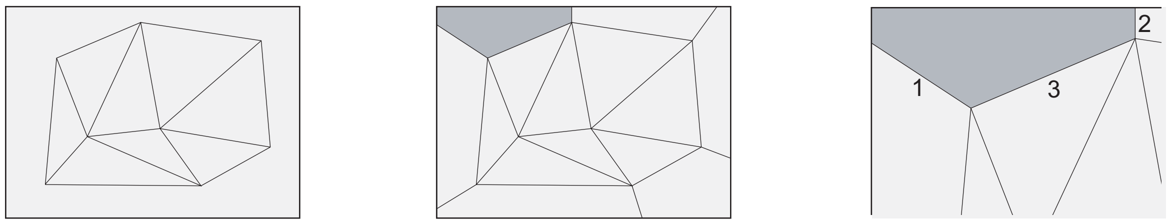


Figure 5.4: Regions outside the convex hull: (left) the Delaunay triangulation only covers points within the convex hull; (centre) the planes from neighbouring triangles on the convex hull intersect at lines which are used to create outer regions like the one highlighted; (right) each outer region is delimited by three lines, the intersection lines, labelled 1 and 2, and the triangle edge, labelled 3. Different intersection lines are produced for mapping x co-ordinates and y co-ordinates.

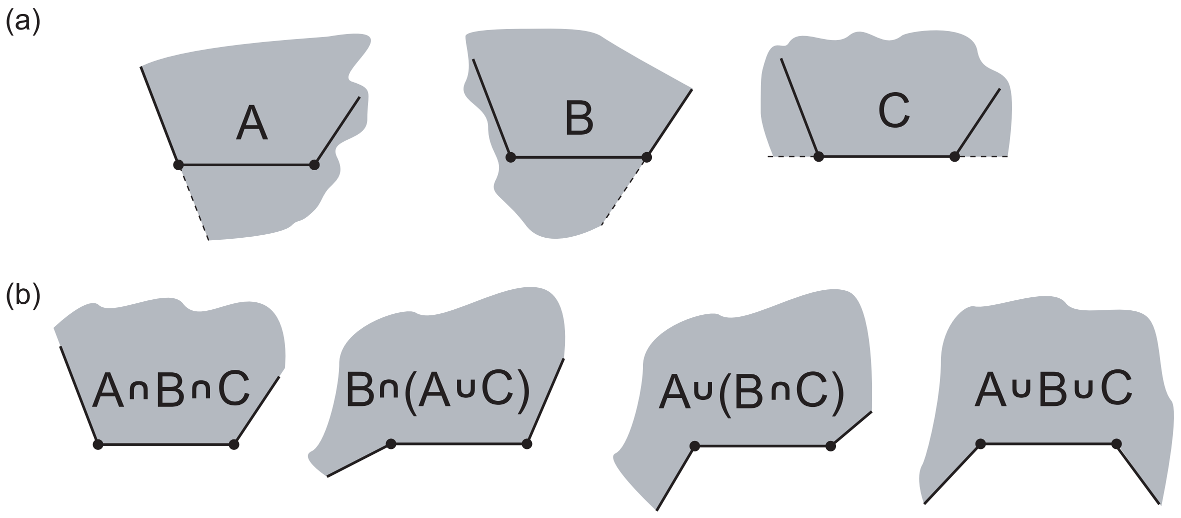


Figure 5.5: Intersecting the half planes that define an outer region: (a) each line defines a half plane; (b) the method by which the half planes are combined, using intersection or union, depends on the angles between the lines. The middle two cases in (b) are analogous when the A and B lines are interchanged.

of the lines must produce the same framebuffer point when mapped with either of the two neighbouring outer regions. If the x co-ordinate of the point is mapped to X_1 with the first plane and X_2 with the second, we require that $X_1 = X_2$. Similarly for the y co-ordinates, $Y_1 = Y_2$. By combining these equalities with equations (5.1) for the X and Y planes, we get the equations of the intersection lines,

$$(a_1c_2 - a_2c_1)x + (b_1c_2 - b_2c_1)y + (c_2d_1 - c_1d_2) = 0\,, \tag{5.2}$$

$$(e_1g_2 - e_2g_1)x + (f_1g_2 - f_2g_1)y + (g_2h_1 - g_1h_2) = 0\,, \tag{5.3}$$

where a_1 to h_1 are the constants from the first region and a_2 to h_2 are those from the second region. The lines defined by these equations delimit the outer regions (Figure 5.4). When a new pen event arrives that is not in any of the inner regions inside the convex hull, it must be assigned to one of the outer regions. Its co-ordinates are substituted for x and y in the left sides of Equations (5.2) and (5.3), and the signs of the results reveal which side of each line the event is on.

Each outer region is delimited by three lines: the two lines that extend to infinity, and the triangle edge on the convex hull from whose end points the other lines start (Figure 5.4 (right)). The points assigned to the outer region can therefore be found by intersecting three half-planes. Figure 5.5 shows the four ways that the half planes can intersect, depending on the angles between the three lines. After calibration the angles between the lines are known, and the binary test for inclusion in each of the half-planes is performed using Equations (5.2) and (5.3) and a similar equation for the line formed by the triangle edge.

Assigning a new pen event to a region

When a new event is received from the pen it must be mapped to a point in the framebuffer. This requires identifying the inner or outer region in which it is located. I have used a simple linear search through the list of regions with the enhancement that when the correct one is found it is brought to the start of the list, producing a 'most recently used' algorithm. I have assumed that the pen will generally move continuously across the desk surface rather than jumping around, so the correct region should usually be found at the first or second place in the list.

Mapping an event to a framebuffer point

When the correct region is found, Equation (5.1) is used to create the framebuffer point. This is true whether the pen point is in one of the triangular inner regions within the convex hull, or in an outer region that uses the mapping planes from one of the triangles on the hull.

5.1.4 Evaluation of mapping functions

I tested the mapping functions (Projective, BP4, BP6 and PLI) on data taken from the digitizer and ultrasonic pen of the Escritoire's desk display. For each pen, the periphery projector displayed a 16×10 grid of targets (Figure 5.6), and I used the pen to select the points and recorded the corresponding pen-space locations. Each mapping was initialized using a set of four or six points as appropriate, chosen so as to be evenly distributed in desk space. The error for a mapping was computed as the mean distance between the desired position of a mapped point and its actual position after being put through the mapping function. Each mapping was refined by repeatedly adding the point with the highest error to the set used to calibrate the mapping.

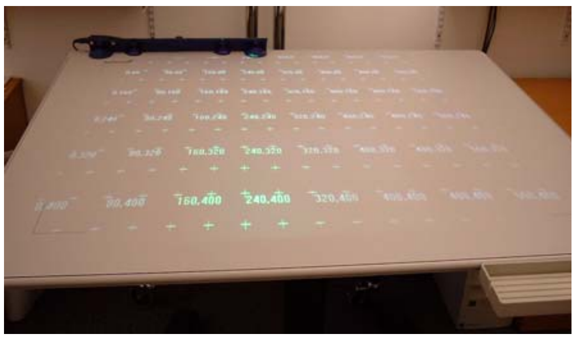


Figure 5.6: I tested the mapping from the digitizer and ultrasonic pens to the projector framebuffer using sets of correspondences between points in the framebuffer and points in pen co-ordinates. The points in pen co-ordinates were obtained by selecting regular a grid of points displayed from the projector.

When describing the calibration of the projectors to the desk I assumed that the transformation experienced by an image as it is projected could be modelled by a projective transformation (Section 4.1.1). Figure 5.7 shows the result of using each of the four mapping functions to calibrate the digitizer to the framebuffer. Projective, BP6 and PLI all converge to an accurate solution, but Projective is the best, achieving the lowest error and obtaining a good solution with just 5 control points. This validates the use of a projective transformation to calibrate the graphics warping.

Figure 5.8 shows the result of using the four mapping functions on data from the ultrasonic pen. PLI performs best: it quickly reduces the error as more control points are added, and converges to a lower error level than the other mapping schemes.

In his account of his work on the Interactive Mural, Guimbretière briefly mentions a system for calibrating an ultrasonic pen to the wall display [Gui02, Section 4.3.1]. The user traces a coarse grid with the pen, then the system uses a 'simple bilinear interpolation method' that is said to improve the uniformity of calibration over the full surface of the screen. No details are given and the accuracy is not quantified. Goshtasby, on whose scheme PLI is based, also experimented with other 2D mapping schemes for image registration [Gos88] including a piecewise cubic interpolating function which, in

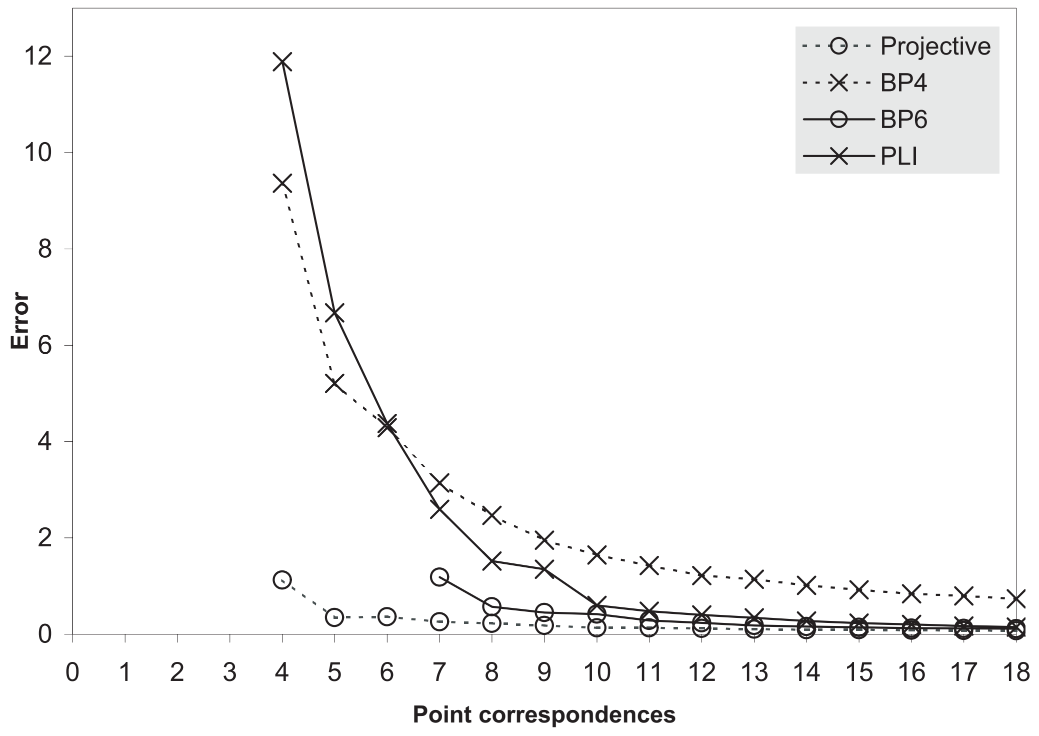


Figure 5.7: Four mappings applied to digitizer data. Average error in pixels is plotted against the number of point correspondences used to initialize the mapping. Projective, BP6 and PLI all converge to a low error. As expected, Projective produces the best results, achieving a low error level with only a few point correspondences.

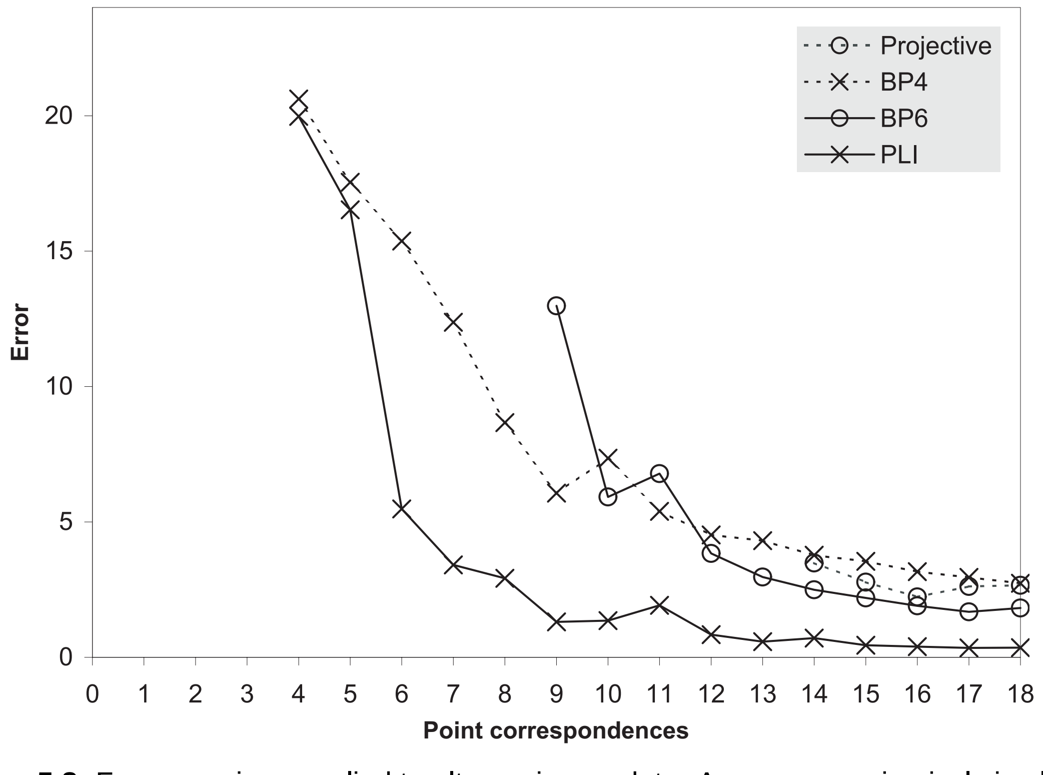


Figure 5.8: Four mappings applied to ultrasonic pen data. Average error in pixels is plotted against number of point correspondences used to initialize the mapping. The error from PLI reduces rapidly as correspondences are added, and it achieves the lowest level.

addition to being continuous, also has a first derivative that is continuous. Although this may avoid visible artifacts in image warping I believe the extra complexity is unwarranted for pen calibration.

5.2 3D input

In addition to the two pens that are used on the desk, the Escritoire provides a *wand* for moving items on the desk and wall displays. The wand is a plastic tube that contains a Polhemus receiver and two buttons (Figure 3.3 on page 49). The Polhemus provides 6 degree of freedom (6DOF) data that specify the location and direction of the receiver in the form of distances, x, y, and z, and angles, azimuth, elevation, and roll. To calibrate a display on the wall the projector displays targets and the user selects them with the wand in a similar manner as for the 2D pens (as described in Section 5.1). During preliminary work on the wall display (Section 3.2) I placed the wand against the wall to obtain 3D points that were used to determine the location of the wall and hence calibrate the system. This method cannot be used for the desk display because the Polhemus uses magnetism to track the position and orientation of its receiver and the metal in the desk distorts the magnetic field. In any case, it may be difficult to get to the targets projected onto a large wall display because they are too high above the ground or because there is an object such as a desk between the user and the plane. Another difference between the 3D wands and the 2D pens is that, in contrast to the systematic error present in the ultrasonic pen readings, the errors in the wand readings are random, so the calibration algorithm must take account of this. The method described below allows the wall display to be calibrated at a distance and is robust to random error in the readings.

5.2.1 Calibrating the wand

A mapping is required from 6DOF wand readings to 2D points in the plane of the projected display. Each reading from the wand is used to create two 3-element vectors: $\mathbf{l}$, the location of the Polhemus receiver, and $\mathbf{d}$, its direction as a normalized vector. A series of targets is displayed on the plane by the projector as for the 2D mapping, but in this case it is the 3D location of each target that is required. The user provides a reading from the wand by aiming at the target and pressing the button on the wand. This is repeated several times (Figure 5.9) until the 3D location of the target has been recovered to a predetermined accuracy. A plane is fitted to the 3D locations obtained in this way to get a model of the display surface. New wand events can then be used to generate 3D pen events by intersecting a line from the wand with the plane. The three stages—obtaining 3D points, fitting a plane, and generating new events—are described below.

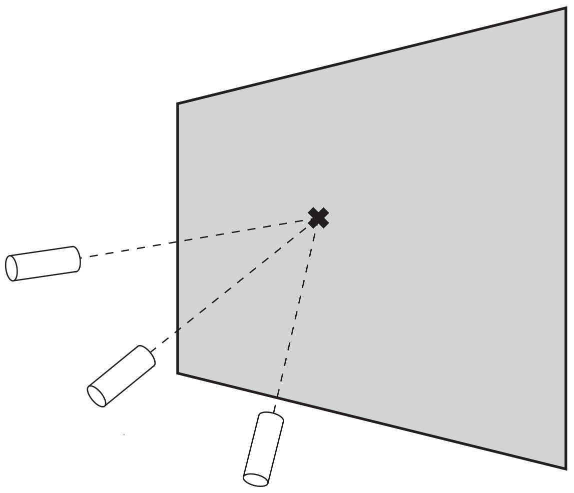


Figure 5.9: To calibrate the wall display a target is displayed on the surface and the user must aim at it from several locations so that an estimate of its 3D location can be obtained.

Obtaining points in the plane

Each of n wand readings can be expressed as an infinitely long line emanating from the tracker, whose points $\mathbf{p}_i$ are defined by

$$\mathbf{p}_i(s) = \mathbf{l}_i + s\,\mathbf{d}_i\ , \tag{5.4}$$

where $i \in [1, n]$ and $s \in \mathbb{R}$. If the n lines are all perfectly aimed at the target they will intersect at a single 3D point. This is unlikely because the user will not have perfect aim, and in any case the tracker readings are subject to random error, so I calculate what I call the *pseudo crossing point*: the point at which the lines are closest.

I find the point where the sum of squared distances to the lines is minimal and use this as the pseudo crossing point. First we note that the centroid of a set of points achieves the minimum sum of squared distances to the points (proof in Appendix A). The centroid $\overline{\mathbf{p}}$ of a set of points on the lines from the trackers is given by,

$$\overline{\mathbf{p}} \quad = \quad \frac{1}{n}\sum \mathbf{p}_i(s_i) \quad = \quad \frac{1}{n}\sum_{i=1}^{n}(\mathbf{l}_i + s_i\,\mathbf{d}_i)\ , \tag{5.5}$$

If all of the lines passed through a single point, given the appropriate values of s_1 to s_n we would get,

$$\overline{\mathbf{p}} \;-\; \mathbf{p}_j(s_j) \;=\; \begin{bmatrix}0\\0\\0\end{bmatrix}\ , \tag{5.6}$$

for each line $\mathbf{p}_j$. Combining Equations (5.5) and (5.6) gives,

$$\sum_{i=1}^{n}(\,\mathbf{l}_i + s_i\,\mathbf{d}_i\,) = n(\,\mathbf{l}_j + s_j\,\mathbf{d}_j\,)\ .$$

We split the vector equation above into its x, y and z components, so for instance the components of $\mathbf{d}_i$ are $\mathbf{d}_{i,x}, \mathbf{d}_{i,y}$ and $\mathbf{d}_{i,z}$, to get three linear constraints on the s_i values. We then stack the constraints for all of the lines to get $3n$ constraints which can be expressed as,

$$\begin{bmatrix}(1-n)\mathbf{d}_{1,x} & \mathbf{d}_{2,x} & \cdots & \mathbf{d}_{n,x}\\ (1-n)\mathbf{d}_{1,y} & \mathbf{d}_{2,y} & \cdots & \mathbf{d}_{n,y}\\ (1-n)\mathbf{d}_{1,z} & \mathbf{d}_{2,z} & \cdots & \mathbf{d}_{n,z}\\ \mathbf{d}_{1,x} & (1-n)\mathbf{d}_{2,x} & \cdots & \mathbf{d}_{n,x}\\ \vdots & \vdots & \ddots & \\ \mathbf{d}_{1,z} & \mathbf{d}_{2,z} & & (1-n)\mathbf{d}_{n,z}\end{bmatrix}\begin{bmatrix}s_1\\ s_2\\ \\ \vdots\\ \\ s_n\end{bmatrix} = \begin{bmatrix}n\mathbf{l}_{1,x} - (\sum_{i=1}^{n}\mathbf{l}_{i,x})\\ n\mathbf{l}_{1,y} - (\sum_{i=1}^{n}\mathbf{l}_{i,y})\\ n\mathbf{l}_{1,z} - (\sum_{i=1}^{n}\mathbf{l}_{i,z})\\ n\mathbf{l}_{2,x} - (\sum_{i=1}^{n}\mathbf{l}_{i,x})\\ \vdots\\ n\mathbf{l}_{n,z} - (\sum_{i=1}^{n}\mathbf{l}_{i,z})\end{bmatrix} = \mathbf{As} = \mathbf{l}\ . \tag{5.7}$$

The $3n \times n$ matrix on the left of (5.7) we will call $\mathbf{A}$, and the vectors we will call s and l respectively. The best value of vector s, the one that minimizes the sum of square errors, is the one that minimizes $||\ \mathbf{As} - \mathbf{l}\ ||$. The desired value of s is given by solving

$$\mathbf{A}^{\top}\mathbf{As} = \mathbf{A}^{\top}\mathbf{l}\ .$$

I solve this using LU decomposition to get s. I then use the centroid of the points generated from s_1 to s_n as the pseudo crossing point. The minimum number of lines necessary to create a pseudo crossing point is 2, but I impose a higher minimum number, say 3 or 4, to get a better location that is based on more samples. Also, the mean distance of the lines from the point

$$\frac{1}{n}\sum_{i=1}^{n} ||\, \overline{\mathbf{p}} - \mathbf{p}_i(s_i)\, || \, ,$$

is tested against a threshold. If it is too high the user is asked to provide another sample with the wand. This continues until the mean distance is low enough. When more than the minimum number of lines is available, the line with the greatest distance from the pseudo crossing point is repeatedly discarded until the threshold is satisfied or there are no lines left to discard. This means that the algorithm is resistant to the outliers that can occur in the data from the magnetic tracker, because an outlying line will be discarded before the valid samples that converge on a crossing point.

Modelling the display plane

Let the minimum number of pseudo crossing points be m. The minimum number of points that is required to define the plane is three, but I use more ($m = 9$) to get a better approximation. If the points are put in a matrix $\mathbf{B}$,

$$\mathbf{B} = \begin{bmatrix} x_0 & y_0 & z_0 \\ x_1 & y_1 & z_1 \\ & \vdots & \\ x_m & y_m & z_m \end{bmatrix} ,$$

then the plane that minimizes the squared distances from the pseudo crossing points is defined by its normal vector $\mathbf{n}$ and distance from the origin d. This is an Orthogonal Least Squares problem [Sim99]. For a plane that passes through the origin the optimal value for $\mathbf{n}$ is the eigenvector corresponding to the smallest eigenvalue of $\mathbf{B}^\top\mathbf{B}$. In general the plane will not pass through the origin, but it will pass through the centroid of the points which, as for the 2D case above, is the point that minimizes the sum of squared distances to the points. I find the desired plane by first fitting a plane to a set of points whose centroid is the origin, by subtracting the centroid $\mathbf{c}$ from the points in $\mathbf{B}$. The normal $\mathbf{n}$ to the plane through the original points will be identical, and the distance of that plane from the origin will be $d = \mathbf{c}^\top\mathbf{n}$, that is, the scalar product of the centroid and the normal.

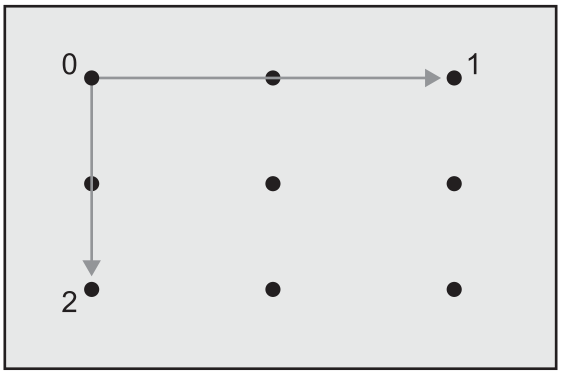


Figure 5.10: Creating a co-ordinate frame on the 3D plane: the 3D locations of targets 0 and 1 from the framebuffer are used to create a horizontal axis on the plane, and those of targets 0 and 2 to create a vertical axis.

Generating 2D events

When a new event from the wand is received, in the form $(\mathbf{l}, \mathbf{d})$, a 2D pen event must be generated. While the co-ordinate system on the plane is arbitrary, choosing one that is aligned with the projected graphics makes the system easier to conceptualize. My system chooses three of the projected targets that define an orthogonal basis in the framebuffer where the first acts as the origin, the second defines the x-axis and the third defines the y-axis (Figure 5.10). The 3D approximations to these points, which are obtained as described above, are projected orthogonally onto the plane and used to create two perpendicular 3D vectors x and y in the plane that define a co-ordinate space.

An event from the wand, with its 6 degrees of freedom, must be converted into a 2D point. As previously mentioned in Section 3.2 this could be achieved by projecting the location l orthogonally onto the plane, or alternatively by using the intersection of the line defined by the wand with the plane (Figure 3.4 on page 49). For the first method the 2D point is solely based on l, and d is not used. Accuracy is not determined by distance from the plane, and putting the wand close to the plane results in it acting like the 2D pen input devices. For the second method we find a point on the line through the wand, $\mathbf{p}(s) = \mathbf{l} + s\mathbf{d}$, that is also in the plane, thus $\mathbf{n}.\mathbf{p}(s) = d$, and rearranging,

$$s = \frac{d - \mathbf{n}.\mathbf{l}}{\mathbf{n}.\mathbf{d}} \ .$$

The 3D point on the line can then be converted to a 2D location by projecting onto the plane in the same way as in the first method. Negative values of s should be discarded because they correspond to the half of the line that points backward out of the wand.

5.2.2 Evaluation of 3D Input

When selecting projected targets with the wand, the user must aim from a different location to generate each reading, and ideally will maximize the angles between the directions of the readings for any single target. This an important point to convey to any new user. Of the two methods for generating 2D events—orthogonal projection or intersection of a line with the plane—the second was accepted much more readily by users, as explained in Section 3.2. The second method has proved to be much better, and although it relies on precision in the angle at which the wand is held, which the first method does not, feedback via a cursor allows a user to select items on a wall 2 metres away and move them to the desk, even when the calibration is not very accurate. The use of a magnetic tracker does place limitations on the physical configuration of the system because the transmitter and receiver must not be placed next to metal objects, and especially not near CRT monitors because of the strong magnetic field they generate. 3D ultrasonic trackers are now available, such as the Intersense MiniTrax [MT] that is strapped to the back of the user's hand so that a pen could be held at the same time. Such a device could perform the same function as the wands without the problems of magnetic interference, although line of sight is required for these systems.

5.3 Summary

I have calibrated the pen input devices to the projected display by obtaining mappings from pen co-ordinate spaces to the framebuffers of the projectors, and I have defined a primary input device for each interactive surface which defines how the projected graphics are warped as described in Chapter 4. If the general form of the required mapping is known a global mapping function can be used, such as a similarity, affine, or projective transformation, or a bivariate polynomial mapping, which may have some terms omitted. Otherwise a local mapping function is better. This type of mapping need not be supported by the graphics hardware if it is only used to calibrate secondary input devices. I have presented a local mapping function called Piecewise Linear Interpolation, which I have adapted from a method for image registration. My experimental data show that a projective mapping can accurately calibrate the Escritoire's digitizer pen, and that PLI is better for the ultrasonic pen.

I have added a projected wall display to the Escritoire to provide even more display space, and interaction with this is via the 3D wand devices. I have given a method to calibrate this display that is robust to the random error in the readings from the tracking device and that allows the user to calibrate the display from a distance so that surfaces that are out of reach can still be used to hold information. A large visible cursor helps hand-eye co-ordination and makes it easy to move items around on the wall or between wall and desk.

Chapter 6

System Architecture

The low-level, performance-dependent code that implements the co-ordinate space transformations and graphics processing described in Chapters 4 and 5 is very different from the device-independent code and data needed to produce the sheets of virtual paper on the desk. I have separated these two parts to create a client and a server. This helps keep the roles of the two pieces of code separate so they can be designed and optimized differently, and also allows multiple clients to be connected to a single server to support synchronous distributed collaboration.

This chapter first explains the division of labour between client and server, and the protocol used to inform each one of changes that occur on the other. It then describes the server, including the various items that have been implemented to run on the server and displayed on the desk. Finally, it mentions some points that are pertinent to the programming of the client.

6.1 Client-server design

The Escritoire is a client-server system. The sections below describe the roles assigned to the client and to the server, the protocol of messages used to exchange information between them, and a particular point about the protocol: the choice between client pull and server push, that is, the choice between the client continually requesting new information from the server and the server sending new information to the client as soon as it is available.

6.1.1 Design overview

The client program runs on the computer that has the projectors and various input devices connected to it. The server program can run on any computer accessible from the client via an IP network. The design and protocol were inspired by the X Window System [Nye90], although in X it is the *server* that runs on the computer with the input and output devices, and the *client* that implements the application program. The difference arises because with the X Window System a single user can interact with applications on multiple computers, but with the Escritoire multiple users can connect to a single computer to work collaboratively on the same data. In both cases there is one server and multiple clients that connect to it.

There is a clear distinction between the design criteria of the Escritoire's client and server programs. The purpose of the client program is to process messages from the server and update the images displayed through the projectors as quickly as possible. It depends on the hardware available on the client computer to receive input events and perform graphics processing. The client program also relays input events to the server. The server program reacts to input events and to application program events, and sends updates to the client when necessary. The differences between client and server are summarized in Table 6.1.

	Client	Server
control flow	sequential	event driven
programming language	C++	Java
system dependence	dependent	independent
state storage	stateless	stateful

Table 6.1: The different characteristics of the Escritoire's client and server programs: the client is dependent on the hardware of the machine on which it runs, and is designed for speed of execution; the server is hardware-independent.

I have kept the structure of the client program simple, and it uses just two threads. The first thread processes standard program signals from the operating system, obtains new events from the input devices, and maps each event to a desk-space location before sending an appropriate message to the server. The second thread receives messages from the server, makes the necessary changes to the state of the client, and refreshes the projected displays using the method described in Section 4.2. The low-level programming required for the client program makes it more susceptible to bugs, and makes the effects of those bugs more severe, but because all the state of the system is held in the server the client can be restarted at any time, which makes it possible to recover from serious errors and assists debugging. The reporting of locally generated events to the server is kept separate from the presentation of the effects of events, which is determined by messages received from the server. This means there is no difference in the way that the effects of locally and remotely generated events are processed. This is desirable because a design that does treat the effects of these two types of event differently will be more complex and has potential for bugs in the mechanism by which the effects are combined, as found by Greenberg *et al.* [GRWB92b].

The server has a more complex structure than the client, but the coding and design are simplified because it is written in Java rather than C++ and is independent of the hardware on which it is run. The server maintains *portfolios* which contain the executable code that implements the sheets of virtual paper on the Escritoire's desk display. The client is only required to display *tiles*, which are the views of the portfolios that are presented to the user. The server manages the portfolios, and sends messages to the client instructing it to create, move, update, re-order, and destroy tiles, so that the client maintains a consistent view of the contents of the desk. The relationship between the tiles on the client and the portfolios on the server is depicted in Figure 6.1.

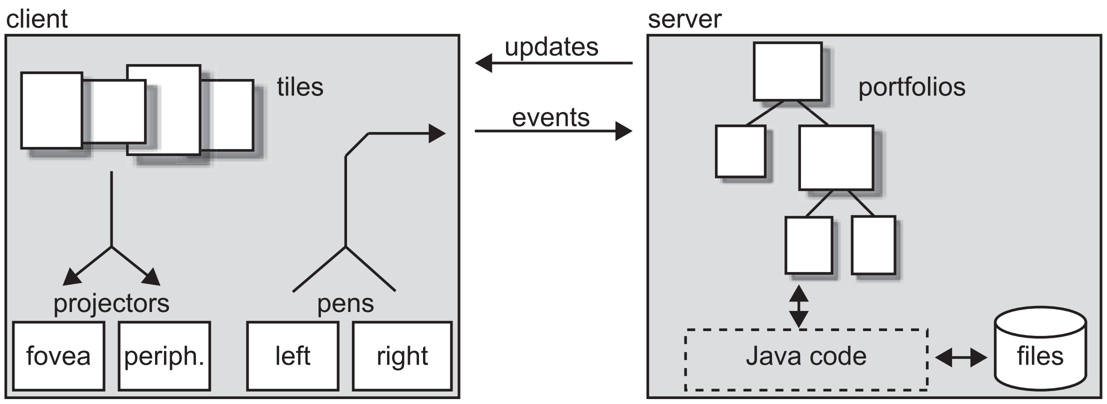


Figure 6.1: The client and server of the Escritoire. The client is implemented with just two threads: one receives messages from the server, updates the ordered list of tiles, and prepares the output for the projectors, while the other processes input from the pens and sends messages to the server. The server holds the portfolios which receive events and generate updates, and saves their state to disk.

6.1.2 Protocol

The client and server communicate using a protocol implemented over TCP sockets. In this interactive system small event messages must be delivered promptly, so Nagle's algorithm [Nag84] for delaying TCP transmissions to allow packets to be merged is disabled using the TCP_NODELAY socket option. Messages are sent in a binary format, where each message consists of a code that identifies the type of the message, followed by a fixed or variable amount of data appropriate to that message type.

The client initiates the connection, then sends a `hello` message to the server. The server responds with a `join` message. After that, the client continually sends input event messages to the server, and the server responds with messages that are used by the client to update its display. The messages available to client and server are listed below, starting with the messages that can be sent from client to server. Each message type is accompanied by a description of its parameters, which are all 32-bit integers (ints), 32-bit floating point numbers (floats), or arrays of 32-bit values.

`hello` Informs the server that a new client has joined. Both client and server support multiple display surfaces so the client specifies the numbers of the displays to which it wishes to connect. After sending the message the client will wait for a `join` message and information about all tiles.

- array size: int indicating the number of elements in the array.
- display numbers: variable length array of ints.

`keyboard` Signals a keyboard event. Specifies which key it was and whether that key was pressed or released.

- key code: int.
- down: boolean indicating if key was pressed.

`pen`

Signals a change in state for one of the pens. Specifies the pen type because a single user has multiple pens, and the current state of the that pen. Because there can be multiple display surfaces the state includes a display number. The server will infer pen movements and button presses by comparing the new pen state with the state from the previous message.

- pen type: int.
- display: int.
- x co-ordinate: float.
- y co-ordinate: float.
- button press bit field: indicates whether each of the buttons on the pen is pressed.

`ready`

Indicates that the client is ready to receive new messages. The server will respond by sending all buffered messages, followed by a `burst terminator`. This message has no fields.

The types of messages sent from server to client, with their parameters, are listed below.

`join`

Sent to the client in response to a `hello` message. The client is sent the resolution that the server is using for the tiles so it can use the same resolution for the fovea texture to avoid scaling the tile images. This message will be followed by details and updates for all tiles.

- resolution: float, in pixels per metre.

`create tile`

Causes the client to create a new tile with a specified id number.

- tile number: int.
- desk-space width: float.
- desk-space height: float.
- pixel width: int.
- pixel height: int.

`move tile`

Move a tile to a new location.

- tile number: int.
- display: int.
- x co-ordinate: float.
- y co-ordinate: float.

`update tile`

Replace a rectangular region of the bitmapped image of a tile. The new pixel data are sent in a variable length message and are used to replace the specified rectangular portion of the tile.

- tile number: int.
- x: int.
- y: int.
- width: int.
- height: int.
- pixel data: length is dependent on the width and height values.

`destroy tile`	Causes the client to delete an existing tile. • tile number: int.
`order tiles`	Presents the client with an array of tile id numbers for the order of the tiles on one of the display surfaces. This determines the way in which tiles occlude each other when they overlap. • display: int. • array size: int indicating the number of elements in the array. • tile numbers: variable length array of tile id numbers.
`cursor`	Reports the new location of one of the pen cursors. • cursor id: int. • display: int. • x co-ordinate: float. • y co-ordinate: float.
`burst terminator`	This is added to the end of each series of messages sent by the server. The message has no fields.

The locations of tiles and pens are specified using an integer and two floating point numbers which specify the display surface that the location is on—because there may be multiple surfaces corresponding to desk and wall displays—and the desk-space co-ordinates within that display surface. The server sends one `order tiles` message for each display surface to indicate which tiles should appear on top of which others. The way that the tiles are actually drawn was described in Section 4.2.3. The client uses the `cursor` messages from the server to draw pen traces, as described in Section 7.2

6.1.3 Client pull versus server push

The Java implementations of the portfolios on the server can create new threads and run concurrently with each other and asynchronously with the message handling thread on the server. Also, there may be other users connected to the server that are interacting with the portfolios, so new messages may be generated at any time. This suggests using a server-push scheme whereby as soon as a new message is created at the server it is transmitted to the client. However I have assumed that the client will normally be the bottleneck in the system because of the intensive processing necessary to achieve a high frame rate on multiple display devices. Forcing the client to buffer messages while it is performing this processing is undesirable, and if they are buffered at the server there is potential for coalescing multiple messages into a single one, thus saving network capacity and processing at the client. VNC, a remote display system described below in Section 6.2.1, uses a client-pull approach to get an adaptive quality [VNC, RSFWH98]: when a client becomes overloaded it makes less frequent requests for updates. The Escritoire could use client pull to avoid overloading the client, but achieving the necessary fast responses to new messages generated at the server would require continuous polling which would waste network capacity.

The solution I have employed is to switch between server push and client pull automatically. The message traffic between client and server is characterized by periods of inactivity when the user is thinking, punctuated by bursts of message data when the user is performing an action. During a burst of activity the client repeatedly polls the

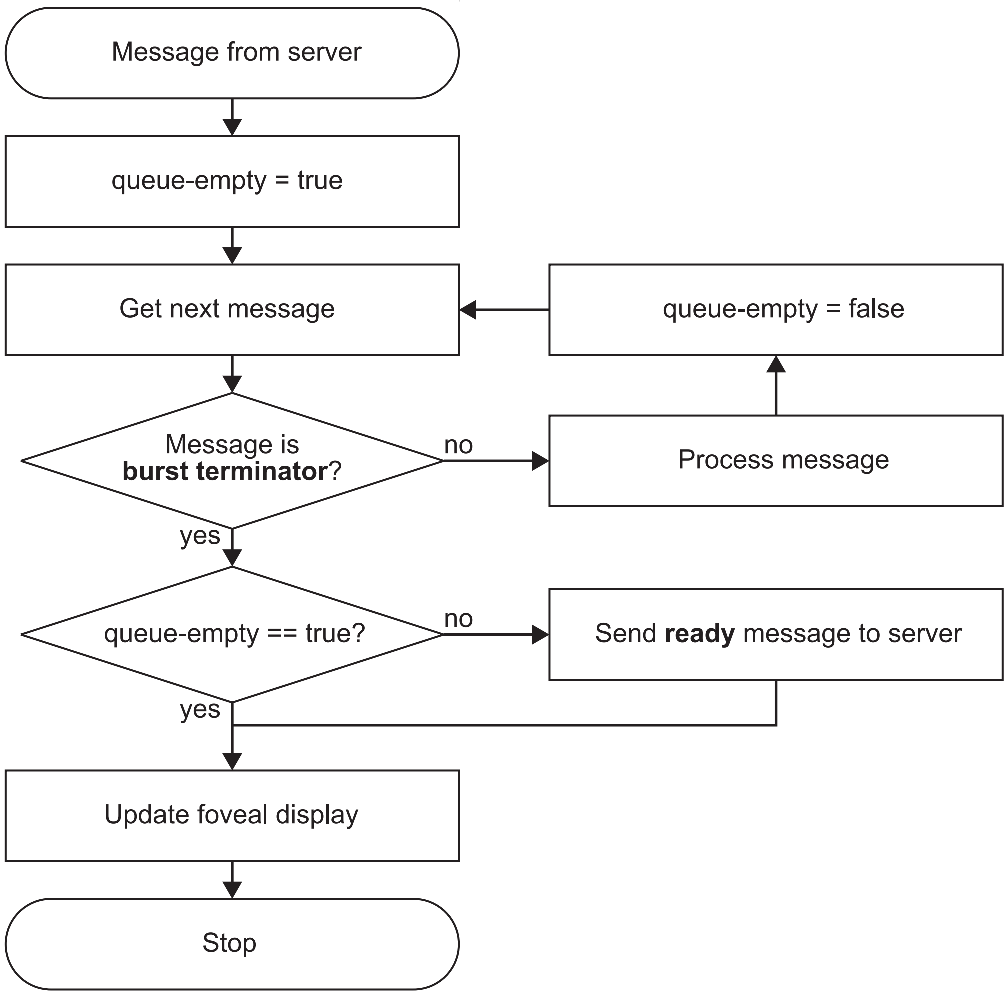


Figure 6.2: Receiving messages at the client. The messages received from the server are processed in sequence until a `burst terminator` appears. If multiple messages were received a `ready` message is sent before the display is updated to request more messages from the server—client-pull mode. If just the burst terminator was received, no `ready` message is sent—server-push mode.

server for new messages then updates its displays using the new data, thus a client-pull scheme is used. The server responds to each of the the client's `ready` messages with a burst of messages ended by a `burst terminator`. When the server has no more messages the burst contains only the `burst terminator`, and the client then switches to server push mode. The server will send a burst as soon as it has a new message.

Figure 6.2 shows the procedure that the client uses, which consists of receiving bursts of messages and refreshing the projected displays at the end of each burst. When a message is received from the server the client processes it and all subsequent messages until a `burst terminator` is received. A boolean variable called queue-empty then indicates whether the burst actually contained any messages apart from the terminator. If it did, a `ready` message is sent to the server before the final step of updating the foveal display. While updating the display the client will be occupied so the `ready` message is sent first, which provides time for it to reach the server and for the subsequent burst of messages from the server to be sent, and buffered at the client. They will be processed when the client has finished updating. A burst containing no messages indicates that the server has no more data to send, so the system is then in server-push mode and the client does not send a `ready` message.

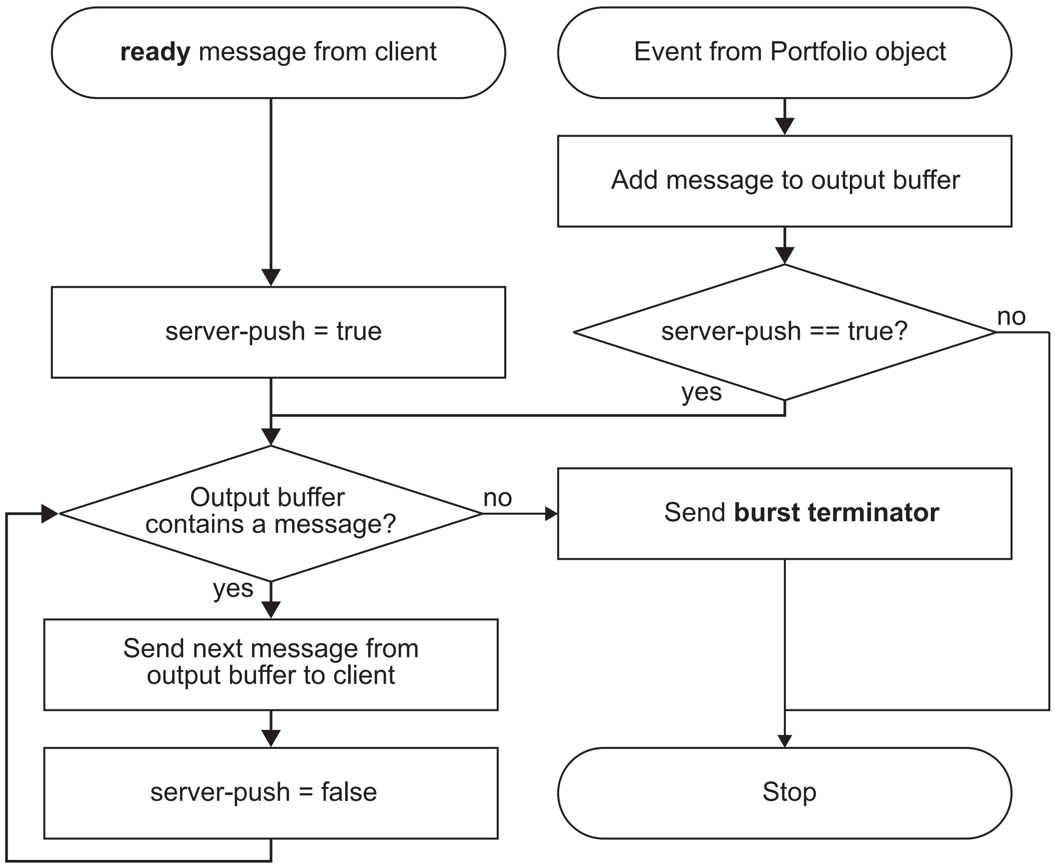

ready message from client
Event from Portfolio object
Add message to output buffer
server-push = true
server-push == true?
no
yes
Output buffer contains a message?
no
Send **burst terminator**
yes
Send next message from output buffer to client
server-push = false
Stop

Figure 6.3: Sending messages from the server. When a `ready` message is received from the client, all messages in the output buffer are sent, followed by a `burst terminator`. If the buffer was empty the server enters server-push mode. When a Portfolio triggers a change that requires a message to be sent to the client, the message is added to the output buffer, then if the server is in server-push mode it sends all buffered messages to the client.

Figure 6.3 shows the procedure that the server uses to respond to a `ready` message from the client or to an event generated by one of the portfolios. A `ready` message causes all buffered messages to be sent to the client, followed by a `burst terminator`. If there were no buffered messages, a boolean variable called server-push is now true, indicating that the next new message from a portfolio should be sent straight to the client rather than buffered. When successive messages are buffered there is potential for coalescing them. The coalescing of buffered messages is described below in Section 6.2.3.

6.2 Escritoire server

The server program is event-driven: its main task is to monitor a queue of incoming events and process them sequentially. It is not dependent on any operating system or hardware devices because the client deals with all system-dependent components. This leaves the server free to deal with the complexities of loading, updating, and saving the portfolios that are the programmatic sides of the sheets of virtual paper on the desk.

6.2.1 Portfolios

The server holds a tree of portfolio objects. The Java classes for these objects are all derived from a class called *Portfolio*. This class contains code to perform general tasks such as maintaining a list of the portfolio's children in the tree and passing events that are not handled by the portfolio down to its children.

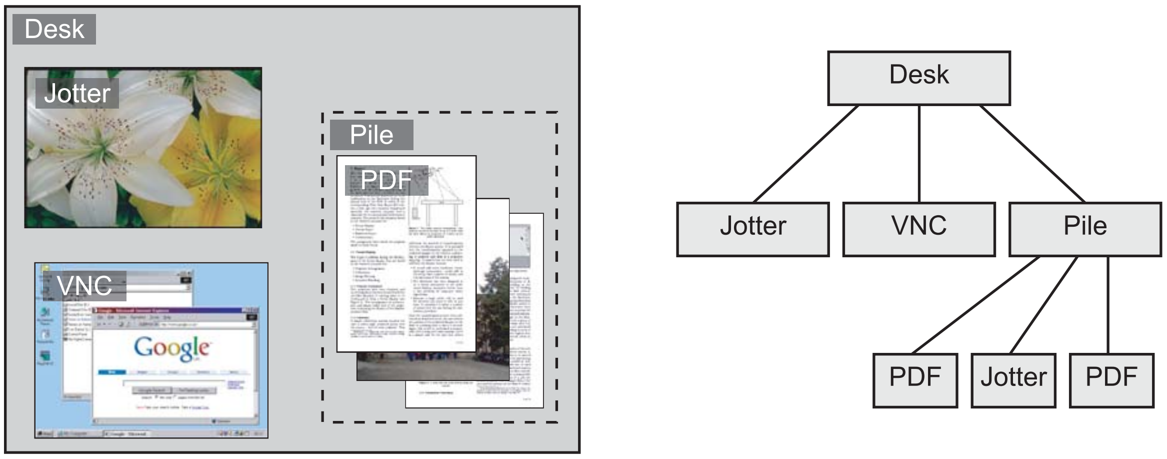


Figure 6.4: The tree of portfolio objects: (left) sheets of virtual paper are arranged on the desk; (right) the corresponding portfolio objects are held in a hierarchical structure on the server.

Each display surface has a tree of portfolios with a *Desk Portfolio* at the root. *Pile Portfolios* are created by the user to group other portfolios on the desk. I have implemented three other types of portfolio: *Jotter Portfolios* allow a bitmapped image to be viewed and annotated; *PDF Portfolios* allow a PDF document to be viewed and annotated with vector strokes; and *VNC Portfolios* connect to a computer that has a standard desktop metaphor interface, and allow the computer's display to be placed on the desk and controlled with the pen.

The portfolios are stored in a tree structure, where a portfolio's children are contained within its bounds in desk space (Figure 6.4). Pen events are sent to the Desk Portfolio at the root, and from there they may be passed down the tree. As an event is passed down it is translated to give it a desk-space location relative to the origin of the portfolio it is being passed to, as occurs with mouse events in a hierarchy of components for a conventional graphical user interface system such as Java's AWT and Swing libraries [AWT]. Events familiar from conventional window systems are generated. *Press*, *release*, and *click* events are generated from the pen button data. *Move* and *drag* events are generated from the pen location data. When the pen moves from one of a portfolio's children to another, *enter* and *exit* events are generated and passed down to the appropriate children. Finally, *lift* events, which are not found in a conventional window system, are generated when the pen is lifted from the physical surface, and contain the last known position of the pen. This movement is not possible with a mouse because the mouse pointer always has a location somewhere on the screen, but the user of a pen input device will often remove the pen from the surface, and the portfolios on the desk should be made aware of this change. The addition of lift events was prompted by user feedback (Section 7.3.2).

PDF Portfolio

A PDF Portfolio allows a document in Adobe Portable Document Format [PDF] to be placed on the display to be read and annotated (Figure 6.5). Before a PDF document is displayed for the user to read it must be rasterized, that is, the pages must be converted into bitmapped images. The server achieves this by calling Ghostscript [GS] which will generate a series of bitmaps from a multi-page PDF document. The Ghostscript options that are used are given in Appendix C. The document can be annotated with strokes that are stored in a vector format. Since PDF is a vector format, based on the Postscript language, the annotations can be added directly to the file. This is achieved using a Java PDF library [IT] that can append the annotations to the source code of the pages of the PDF document. The PDF document, complete with annotations, can then be reviewed on a conventional workstation using a viewer like Adobe Reader [Ado].

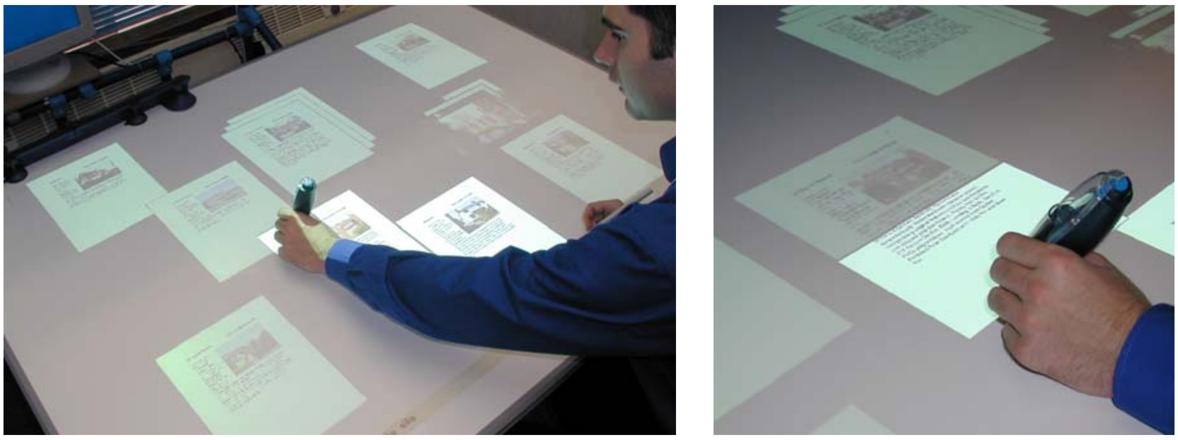


Figure 6.5: PDF Portfolios: (left) many sheets of PDF are arranged on the desk surface; (right) the user can bring any sheet to the fovea to view it in detail.

Items such as presentations and word processed documents created in standard application programs can be converted to PDF documents in that same way that they are printed to paper because printer drivers are available that output PDF. The affordances of paper documents have not been supplanted by current application programs, which means physical paper is still used for many tasks [JJK$^+$93]. A study comparing reading of paper and on-line documents for the purpose of summarization [OS97] found that the benefits of paper outweigh those of computerized reading methods. Paper's support for annotation while reading, quick navigation, and flexibility of spatial layout are identified as its major advantages. The study suggests the use of a larger screen, and a display that is a portal onto a large virtual workspace with an overview that shows the entire contents of the workspace. However, the lack of a spatial continuum between the focus of attention and the periphery is given as a disadvantage of this method over the continuous display of a large table with physical papers on it. Conventional monitors seem unsuitable for reading tasks involving many documents, but the extremely large surface offered by a desk-sized display provides the space to accommodate the natural spatial layout of multiple documents.

VNC Portfolio

Virtual Network Computing (VNC) is a remote display system which allows a conventional computer desktop environment to be viewed and controlled from some other computer on the Internet [VNC]. It assumes a standard workstation model of a bitmapped display for output, and a keyboard and multi-button pointing device for input [RSFWH98]. VNC servers that export conventional computer displays are available for many operating systems. The design puts most of the complexity in the server, which has allowed clients to be written for many platforms including a Java client that runs in a web browser.

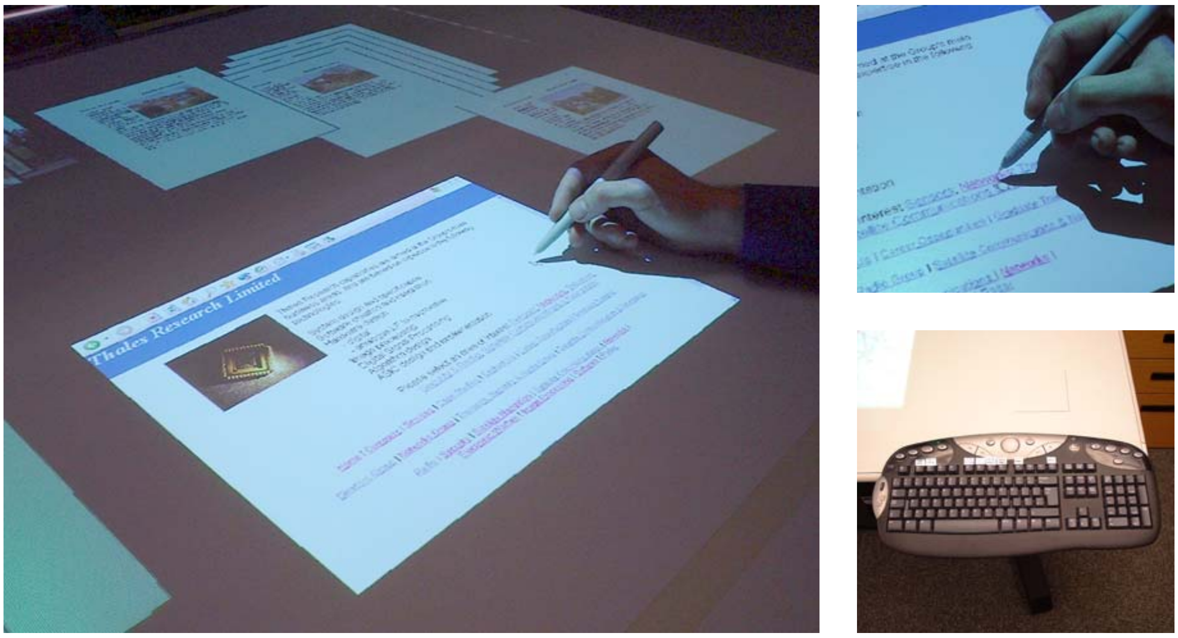


Figure 6.6: VNC Portfolios: (left) a VNC Portfolio allows unmodified conventional programs such as a web browser to be placed on the desk; (top right) the pen in the dominant hand emulates the mouse so the user can click on links; (bottom right) a keyboard is available for text entry.

A VNC Portfolio on the Escritoire is a modified Java VNC client. It converts events from the pen in the dominant hand to mouse events which are then sent to the VNC server, and it relays graphical updates from the VNC server to the Escritoire client so the tile that is showing the display can be updated (Figure 6.6). The VNC Portfolio allows a conventional computer interface to be combined with the desk display, and in fact multiple computers can be used simultaneously. Text entry is necessary for these conventional interfaces so a keyboard is provided (Figure 6.6 (bottom right)). Keyboard events are sent to the portfolio that the dominant pen was last over.

Initially computers were controlled textually from the command line. Then the desktop metaphor became the interface of choice. Not only was it richer and easier to use in many cases, but it also encompassed the old command line. Terminal windows allowed the command line programs to be used alongside ones with graphical user interfaces. A large desk display can encompass conventional window systems, and the command line windows inside them, by having multiple graphical displays on the desk at once, as shown above. Projects such as the Task Gallery, which is described in Section 2.1.1, also embed conventional graphical user interfaces in a new environment, but they provide only an incremental enhancement to the standard interface. The Task Gallery allows more programs than usual to be handled using a small screen. It aims to make it easier to switch between workspaces by providing cues to the tasks they represent, and easier to switch between windows within those workspaces by dynamically arranging the windows. An interface that exploits the large display devices that will become available in future will have to present a new interaction paradigm, and also support legacy applications. The method, described here, of placing conventional graphical displays alongside documents on the Escritoire's desk display shows how this can be achieved.

Jotter Portfolio

A Jotter Portfolio allows a bitmapped image to be placed on the Escritoire's display and annotated with the pen in the dominant hand (Figure 6.7). Annotation strokes are stored in a vector format which allows them to be erased easily by a user, but they are converted to raster form when the image is saved.

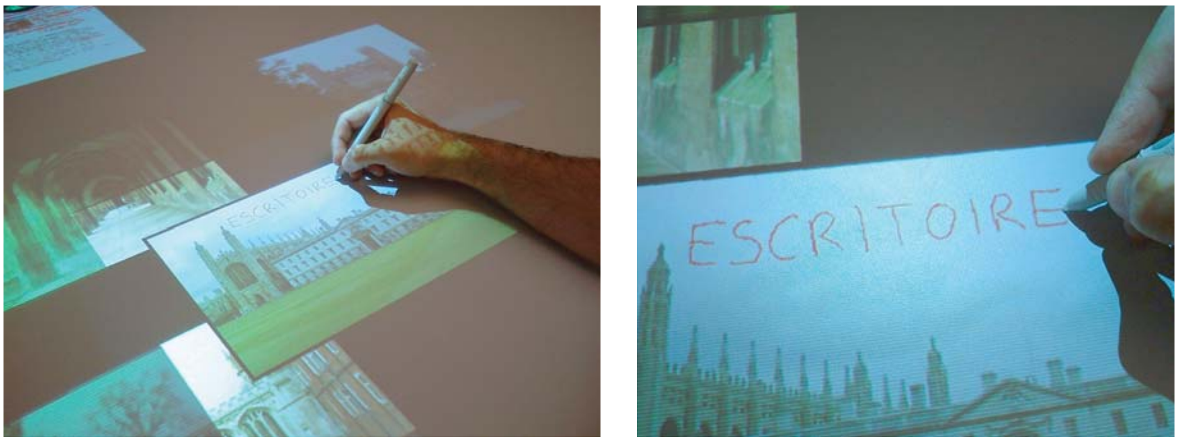


Figure 6.7: Jotter Portfolios: (left) various images are arranged on the desk; (right) the user can annotate any one of them with the pen.

Pile Portfolio

A Pile Portfolio groups several child portfolios so they take less space on the display surface, so they can be moved together, and to allow them to be associated with one another in a particular order (Figure 6.8). A Pile Portfolio processes some pen events so that the user can browse through the pile, move items within the pile, and remove items from the pile. It passes the other events onto the child portfolios. The motivation, implementation, and results of this addition to the interface are described in Chapter 7.

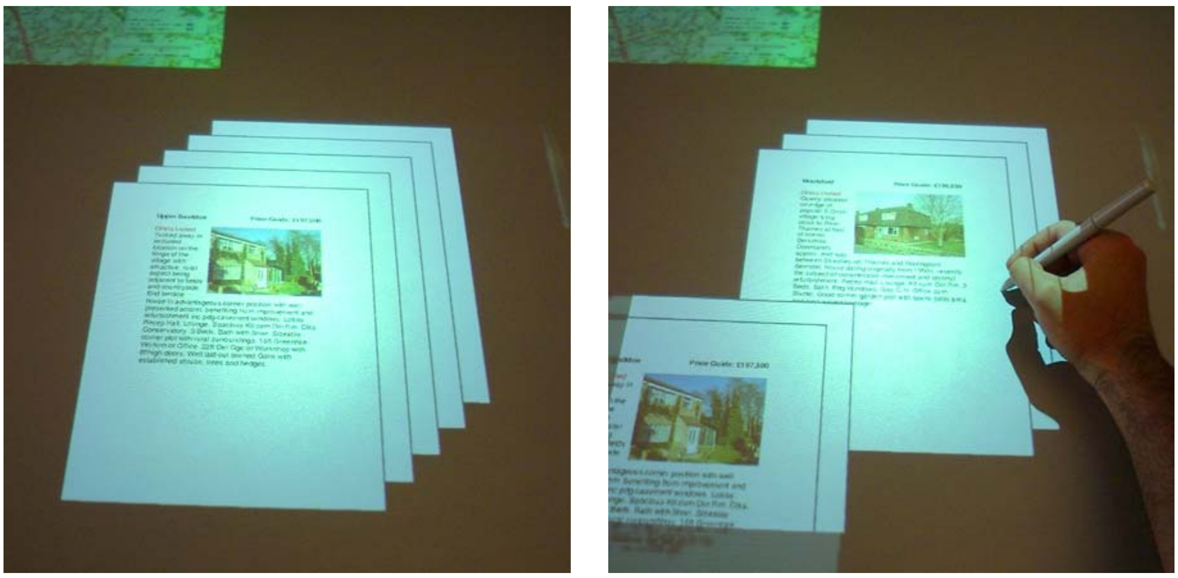


Figure 6.8: Pile Portfolios: (left) a pile of PDF pages; (right) as the user moves the pen over the edges of the sheets the pile splits open to allow browsing.

Desk Portfolio

The Desk Portfolio is the root of the tree of portfolios. There is one for each physical display surface, for instance there may be one for the desk, and a second for the wall. The Desk Portfolio allows its children to be dragged around the surface by appropriate pen events, and passes all other events down to its child portfolios.

6.2.2 Storing application state

Each portfolio on the Escritoire server has a corresponding directory that contains the data that retain its state when the server program is not active. Human readable files store general data, such as the portfolio's location in desk space and its children in the hierarchy, and also type-specific data, such as the name of a PDF or image file. The directory also stores binary files, such as the PDF and image files themselves, and temporary files that the portfolio implementation creates whenever necessary. No state is stored on the client. This makes the client simpler, allows it to be restarted at any time which aids debugging, and avoids problems with conflicting data at different clients and the protocol that would be necessary to avoid such conflicts.

The Escritoire presents a document-centric view of tasks to be performed. The program behind each document—the application—is hidden. Document state is not saved using an application command, but is saved and loaded automatically when the server is stopped and started. This is like the situation with physical pieces of paper, where changes like annotations are persistent and do not need to be saved, and where a person is aided in resuming her work by the condition and locations of documents on the desk. Preserving the state of the system between uses in this way was found to be a desirable feature of a computer called the Cannon Cat released in 1987 [Ras00, pages 29–32], and a similar approach was taken with the Apple iBook.

6.2.3 Buffering messages

The protocol between the client and server of the Escritoire is designed to place as much of the complexity as possible in the high-level code in the server rather than the low-level code in the client, and to limit the amount of data that must be transmitted across the network and processed by the client. During a burst of activity, when the link between client and server is in client-pull mode, new messages that become available at the server are buffered until they are requested by the client. When messages are buffered in this way a newly generated message may make an existing message in the buffer partially or wholly obsolete. In this case the new and existing messages can be coalesced into a single one that gives the same result but requires less capacity during transmission and less processing time at the client. The designers of GroupSketch and GroupDraw [GRWB92b] chose to make each node in their systems send out as much information as possible to the others, not worrying about use of capacity on their 10 Mbps local area network, and to buffer and discard information at the client. This is what happens with messages from client to server on the Escritoire, but it would be unacceptable for the messages from server to client, especially when sending the bitmap data for tile updates over a 256 kbps ADSL link.

The server keeps separate buffers for each type of outgoing message to each client that is connected. The potential for coalescing messages is different for the different types: `update tile` messages have high potential for coalescing because if the regions for two updates on the same tile are identical, or nearly so, an update encompassing both of the regions can be used, which will greatly reduce the amount of data transmitted; `create tile` and `destroy tile` messages, however, have very little potential for coalescing because it is only possible in the unlikely situation that a particular tile is created then destroyed in the short period between bursts of messages being sent to the server. The coalescing tactics used by the server for the various message types are listed below.

`join`	Only one message can be in the queue at a time. Any new message will overwrite the one in the buffer, although this should not occur in practice.
`create tile`	Coalescing is not used because it is unlikely that it will be possible as explained above, and in any case it will not give a large benefit because the messages are small and infrequent. The messages are simply queued.
`move tile`	Only the most recent location for each tile is stored. A new location for a tile overwrites any buffered location for that tile.
`update tile`	The new update region is compared with the existing ones in the buffer for that tile to see if it can be combined with any of them. A linear search is made through the buffered regions. For each one, the area of the bounding region of the new and existing regions is calculated and compared to the sum of the areas of the two regions. If the bounding region has the smaller area it replaces the two separate regions (Figure 6.9). The bounding region is then added to the remaining buffered regions by repeating the algorithm, so multiple merging steps can occur.
`destroy tile`	The messages are simply queued, as for `create tile` messages.
`order tiles`	One array of tile numbers is kept for each display surface. A new order for a display overwrites any buffered order for that display.
`cursor`	The messages are queued. They can be very frequent when a pen is being moved so a limit is put on the size of the queue. A threshold can be put on the distance between successive cursor positions: if the distance between a new position and the most recently queued position is too small the new one is discarded.
`burst terminator`	This message is always sent immediately without buffering.

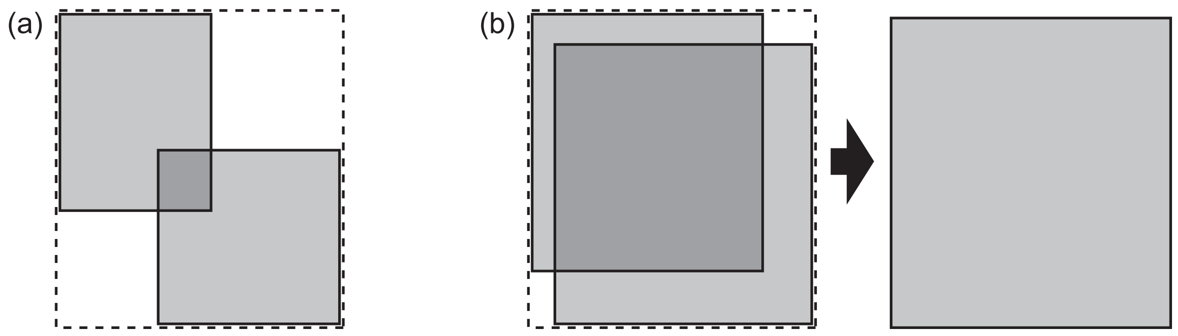


Figure 6.9: Coalescing tile update regions: A pair of update regions for a tile are compared by calculating the sum of the areas of the regions with the area of their bounding rectangle. (a) If the bounding area is larger the regions are kept separate. (b) If the sum of the region areas is larger, the regions are combined.

Buffering and coalescing messages in this way should always be desirable because even if the client can handle the extra message data that would result from having no coalescing, the unnecessary network transmission would be inefficient. In the case of the Escritoire, coalescing `update tile` messages is important because they account for much of the transmitted data, and because updating the textures ready for warping is the most time consuming part of the graphics processing at the client.

6.3 Escritoire client

I have designed the Escritoire client to maximize graphics performance, which allows many tiles to be updated, warped, and displayed on multiple projectors at interactive frame rates. The client also handles input devices.

The client program uses two threads: an output thread that displays graphical output to the user by accepting messages from the server and updating the displays; and an input thread that handles input from the user by accepting events from the input devices, converting them into appropriate messages, and sending the messages to the server. The output thread performs the bulk of the work. It accepts messages from the server of the types listed in Section 6.1.2, alternates between client pull and server push following the flowchart in Section 6.1.3, updates the displays with the new information which involves using the top-down method for display updates described in Section 4.2.3, and warps the graphics ready for projection using the method of Section 4.2.2. During a burst of activity the output thread loops around as quickly as possible, alternating between accepting messages from the server and updating the displays. The performance of the client depends on how fast it can do these two tasks.

The input thread converts events from the 2D and 3D pen input devices into messages which are sent to the server. The messages contain locations in desk space that are generated from the raw 2D or 3D data as described in Chapter 5. Different methods are used to get input events from the three pen input devices. The 3D pen has buttons connected to the computer's parallel port as described in Section 3.2, and position data are obtained by issuing commands to the magnetic tracker through a serial cable. The Mimio ultrasonic pen does not have an open API for retrieving events, but it does have a feature to emulate the mouse. This feature is activated, then the calibration involves finding a mapping from mouse locations to framebuffer points. Wintab [WT] is used to retrieve events from the large digitizer. This is an industry standard API that can be used to obtain button events and accurate position data. Experiences and issues with these input devices are described in Section 7.3.2.

The Escritoire client, when executed on the hardware described in Appendix B, can perform its local processing fast enough to achieve 30 frames per second, as described in Section 4.2.3. In their definition of Information Visualization, Robertson, Card, and Mackinlay [RCM89, page 15] state that a visualization system will ideally achieve 30 FPS. The client-server protocol achieves the desired effect of rapidly transmitting messages to update the tiles and cursors when activity is occurring on the desk display, and logging of network traffic shows that it ceases transmissions when no changes are occurring. In the collaborative tests described in Section 7.4 the capacity of a DSL link was easily able to carry the messages between client and server together with the audio and video channels between the two participants during the main part of the test, but the initial download of the bitmap data for all of the items caused a delay of a few minutes. This could be improved in a future version by reducing the amount of data sent, and by allowing the participants to start working before all of it has been received. On a few occasions there were interruptions in the transmission, presumably due to network congestion. Quality of service guarantees across the network would benefit this, and most other, distributed interactive systems. The round-trip latency of the network transmission was low enough at around 25 ms to make interacting with items on the desk fluid and responsive.

6.4 Summary

The Escritoire is a client-server system. A platform-dependent thin client written in C++ creates the projected graphics and handles the input devices, while an event-driven server program stores the state of the system and processes interactions with the items on the desk. The items held on the server, which I have named portfolios, are implemented by Java classes, and I have described the classes that allow PDF documents, bitmapped images, and conventional programs to be placed on the desk display. I have described the messages that are sent between client and server, and I have given a protocol that switches between client-pull and server-push modes. When the user is interacting with items on the desk the system will be in client-pull mode in which display updates are limited by the performance of the client program, new data is provided when the client requests it, and messages are buffered and coalesced at the server. When the user is thinking, between bursts of activity, the system will be in server-push mode, and any new message will be sent to the client as soon as it is created. I have described methods for coalescing messages of the various types, for instance, `update tile` messages are coalesced by comparing their bounding regions. The client can achieve the desired refresh rate of 30 frames per second when connected to a server over a DSL link.

Chapter 7

User Interface

An A0-size display provides a large amount of space, but this can be filled if many documents or pictures are displayed at once or if the user wants to spread out the pages of a document for perusal. I have avoided the conventional window system technique of iconifying items because it is not consistent with the aim of simulating paper on a real desk, and it hides information. The notion of piles is described below, which I have added to the interface of the Escritoire to save space and allow items to be arranged in groups to convey the relationship between them and to reduce clutter on the desk. The following section describes pen traces, which allow remote collaborators to gesture to each other in the graphical space that they share. The remaining sections of this chapter describe the method and results of the single-user and collaborative tests I have used to analyse and refine the interface of the Escritoire [AR03a, AR03b].

7.1 Piles

In interviews with office workers, Malone [Mal83] found that two important units of desk organization are files and piles. The elements of a file are explicitly titled and arranged in some systematic order, as in a filing cabinet or a conventional electronic filing system. Conversely the elements in a pile are not systematically arranged. The time required to maintain a filing system and the cognitive difficulty of creating appropriate categories for information mean that people often create vaguely classified piles on their desks. This physical arrangement also causes the person to be reminded of tasks to be performed—recognition is easier than recall—and means that the information is easily accessible.

Researchers at Apple Computer also found that employees used piles for informal classification, and that, although an office containing piles often appears disorganized to an outside observer, people knew what was in their piles and could describe their history. They created the *pile metaphor* [MSW92, RM$^+$93b] in which icons representing documents are grouped, and the groups can be used as input or output for an information retrieval system. Using a document vector representation a pile can be split into multiple sub-piles for different subjects by clustering the constituent documents, or a query can be specified by placing some documents in a pile, which defines a combined vector, then asking the system to find more documents on the same subject. Apple has recently patented their pile metaphor [MES+01] and are intending to include it in a forthcoming version of the Macintosh user interface [Orl03]. It will be greatly aided by a file system that automatically stores and updates metadata so that files can be clustered by processing their attributes.

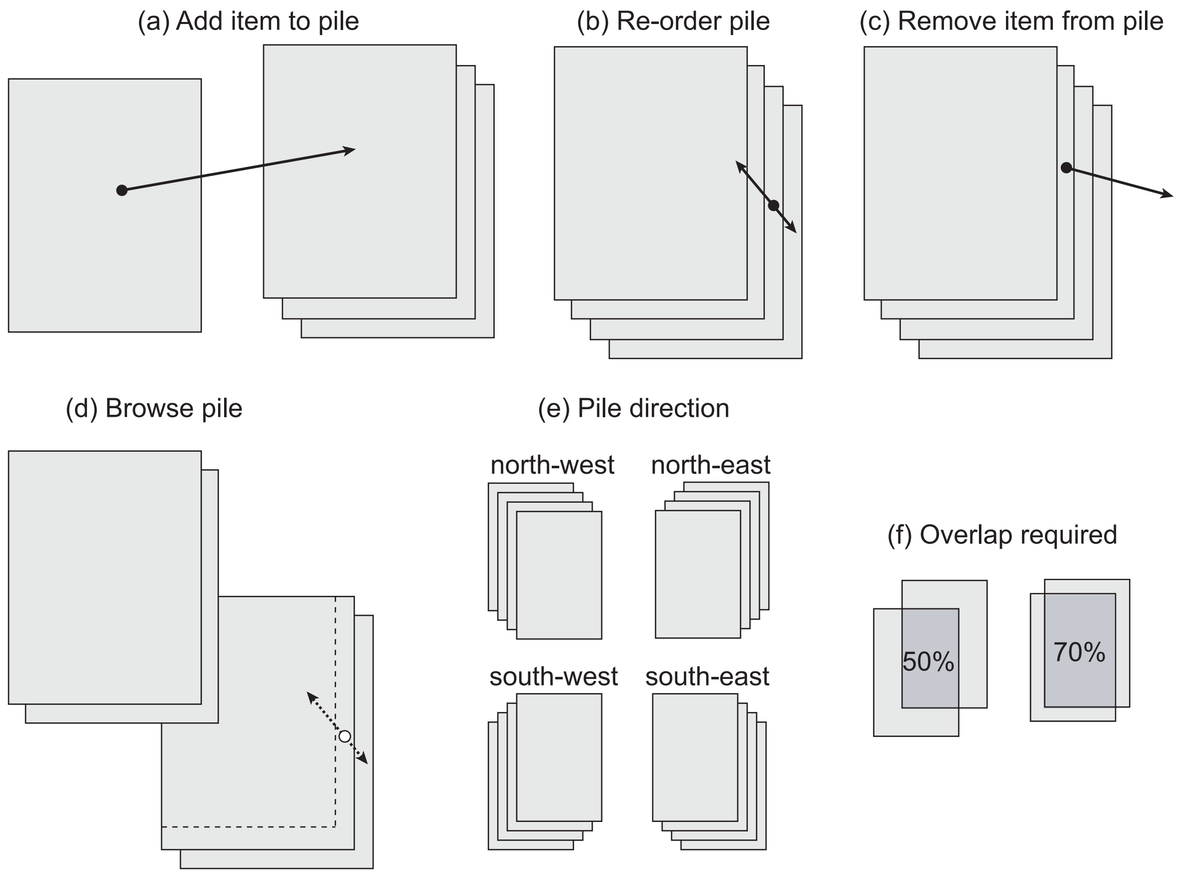


Figure 7.1: Manipulating piles of sheets on the Escritoire: (a) items are dragged onto the top of the pile; (b) items can be dragged to a new position in the pile; (c) items are dragged to remove them from the pile; (d) moving the pen over the pile causes it to split open and reveal the item at that point; (e) one of four pile directions can be chosen. (f) the amount of overlap required for two items to join to form a pile can be set.

I have added the notion of piles to the interface of the Escritoire because piling is an important affordance of physical paper that is not available in conventional user interfaces. Rather than displaying icons representing the documents, the piles hold the actual sheets of virtual paper with their interactive properties unchanged because they are nodes in the tree of portfolio objects stored on the server as explained in Section 6.2.1. The piling feature allows fast and informal grouping and ordering of items on the desk, it extends the set of properties of physical paper that are available through the simulation on the desk, and it also saves space.

When an item is placed on top of another the two join to form a pile, and when an item is placed on top of a pile it joins the pile as the top element (Figure 7.1(a)). Using a dedicated button on the pen in the dominant hand, an item can be dragged to a new position in the pile (Figure 7.1(b)) or it can be removed from the pile completely (Figure 7.1(c)). By placing the pen over the visible edges of the items the user can browse through the pile—it splits open to show the item over which the pen is positioned (Figure 7.1(d)). This splitting action is produced by animating the items in the pile.

The animation is achieved using simple `move tile` messages that are generated by a thread in the Pile Portfolio (Section 6.2.1) object. The items in the pile can be arranged in one of four directions (Figure 7.1(e)). The amount of overlap required between two items before they snap together into a pile (Figure 7.1(f)) must be chosen when the server is configured—I have found that requiring 60 per cent of the area of the smallest item to be covered works well. The users' preference on pile direction and their general reaction to the piles was tested in the experiment described below in Section 7.3.

7.2 Pen traces

When remote users collaborate on a task it is not just the results of their actions that are important to transmit, but also the actions themselves. An early patent [TW80] identified this with a remote lecture system in which the positions on the display where the lecturer was drawing or erasing were continually indicated to the student, allowing him to follow the lecturer's actions. After studying two-person design sessions, Bly [Bly88] concluded that the interactions on a drawing artifact are as important to many design collaborations as the final artifact itself. The conventional view of a shared drawing surface would be that it is merely a medium for creating and storing a drawing, but in studies of small group design sessions Tang [TL88] found that approximately one third of the participants' actions were gestures.

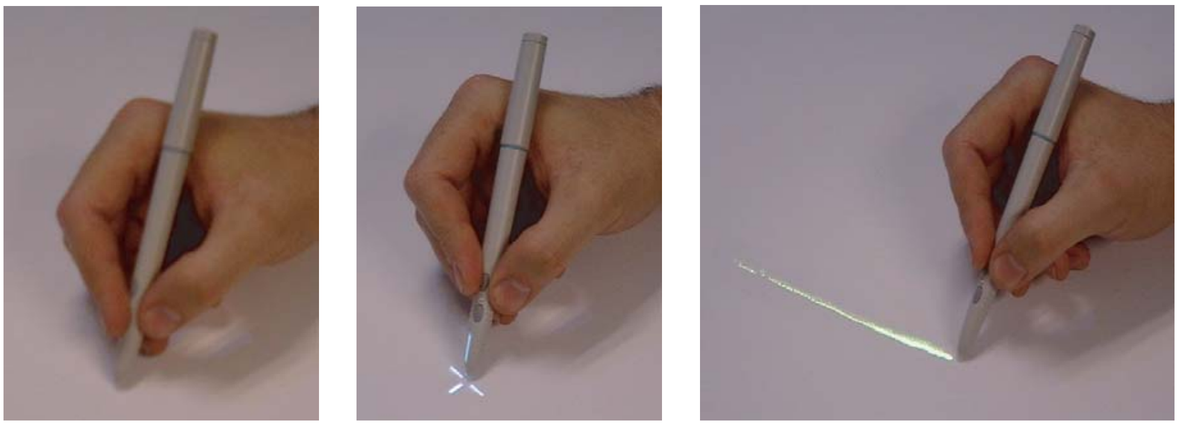


Figure 7.2: The user of the Escritoire can choose from three types of cursor for the local and remote pens: (left) no cursor, (centre) cross hair, and (right) a trace that shows the past motion of the pen.

This motivated the creation of multi-user drawing programs GroupSketch and GroupDraw [GRWB92b, GRWB92a] that use telepointers and present a common view of the work surface. This common view gives continuous fine-grained feedback of the process of drawing which is itself gestural. Design criteria for the cursors in GroupSketch included the following [HPG93]: all cursors within a work surface should always be visible to all participants; cursors must have enough prominence on a multi-cursor display to attract the attention of other participants, cursor movements should appear continuously and with no apparent delay on all displays, and they should maintain the same relative location on every display so that they retain their relation to the work surface objects.

Transient ink that fades away after a few seconds was added to the Designer's Outpost [EKLL03] which is described in Section 2.5.2, in an attempt to allow users to make deictic gestures to each other. About half of the users said that they liked the transient ink, but they felt that having to activate it before using it disrupted the flow of ideas, and one user said he may as well write with permanent ink then erase it afterwards. Telepointer traces that continuously and unobtrusively show the recent positions of remote cursors have been implemented by Gutwin [Gut99] for remote participants working on a graphical task. He has recently performed experiments that show they substantially increase the ability of people to recognize symbols like letters and numbers when they are created as gestures using a telepointer over a network that introduces jitter [Gut02, GP02]. In the experiments participants found it difficult to gesture precisely with a mouse, and it was suggested that a pen input device would have been easier. After observing users of a standard shared whiteboard program, Wolf and Rhyne [WR93] asserted that freehand drawing with a mouse was difficult, but the ease of drawing with a pen makes hand-drawn marks a good solution for gesturing with a pen interface.

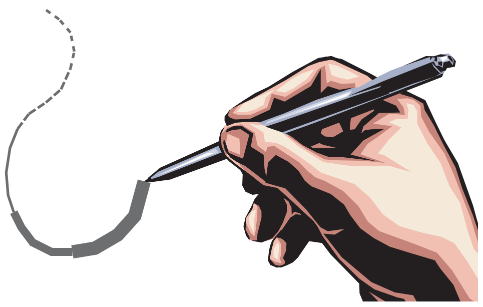


Figure 7.3: I have implemented pen traces by drawing lines between successive pen positions of varying width, and using dashed and dotted lines.

I have added traces to the interface of the Escritoire to allow gesturing between remote users. The user can choose to have no cursors, cross hairs that follow the pens, or traces (Figure 7.2). The `pen` messages from a client are used by the server to generate `cursor` messages (Section 6.1.2) that allow every user to see the locations of the pens of all other users. Each client keeps buffers of time-stamped `cursor` messages that are used to draw traces for a fixed period. I have used a period of one second. Gutwin has implemented traces by drawing lines between successive cursor locations with varying transparency levels, and reports that the transparency consumes significant processing power. I have avoided that problem by implemented them using standard line drawing functions which are very fast—to achieve the appearance of the line fading away, first the width is decreased, then dashed and dotted lines are used (Figure 7.3). Double-buffering is used to prepare the graphics that drive the projectors. Before being flipped to the primary framebuffer the warped graphics are prepared in the secondary framebuffer using the techniques of Section 4.2, then the pen traces are drawn on top—the transformations of Section 4.1.2 are used to convert a pen's sequence of desk-space locations to a sequence of framebuffer locations that are used to draw the traces. The addition of pen traces requires a modification to the client event processing algorithm shown in Figure 6.2 on page 94—instead of only refreshing the display in response to a `burst terminator` from the server, the client also continues to refresh the display while there is a trace that needs to be animated.

7.3 Single-user tests

The interface of the Escritoire at various stages of implementation has been used by visitors to the Computer Laboratory in Cambridge, and I performed informal user tests at Thales Research & Technology who funded my work. The aim of the tests was to get qualitative feedback on the system. The method I used is described below, followed by results from the tests and general observations.

7.3.1 Method

Seven people from Thales took part in the tests. Two were women and five were men, one was left handed and six were right handed, and none had used the Escritoire before. Each participant was first instructed on using both of the pens and given a few minutes to move sheets around on the desk, draw on them, and erase the drawings. Piles were initially disabled. The participant was then asked to perform two tasks to demonstrate the cursor types and the pile system to them, and so that they could experience the interface and form an opinion about it.

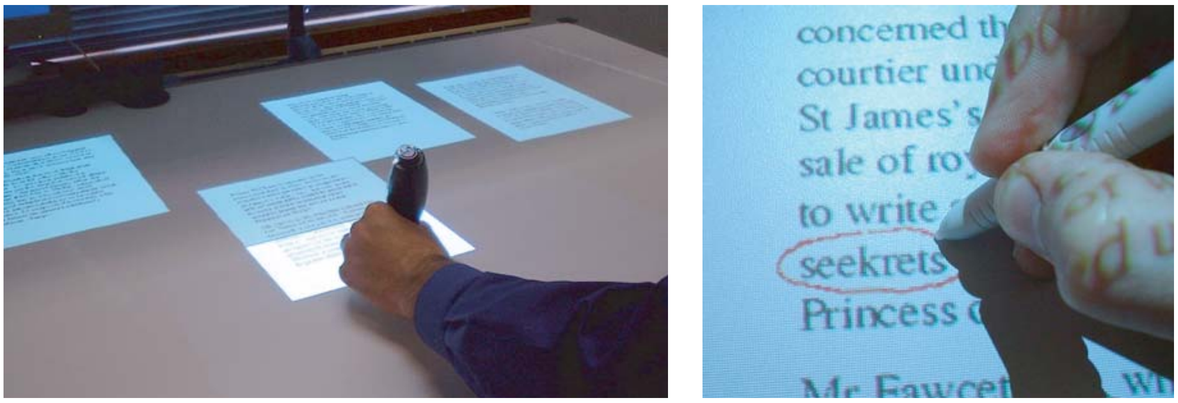


Figure 7.4: The first task the users were given was to highlight the spelling mistake in each of several pages of text: (left) the pages were initially spread over the desk and the user had to bring each one to the fovea; (right) the user then circled the single mistake on each page.

The first task was to highlight the spelling mistakes in four documents consisting of short pieces of text. This was done for each of the three cursor types—none, cross hair, and trace—in a random order, using different documents. The documents were initially placed along the top of the desk so the user had to move each one to the fovea to read it. Figure 7.4 shows the actions involved. Afterwards the user was asked which type of cursor he preferred, and was asked for comments on the cursors.

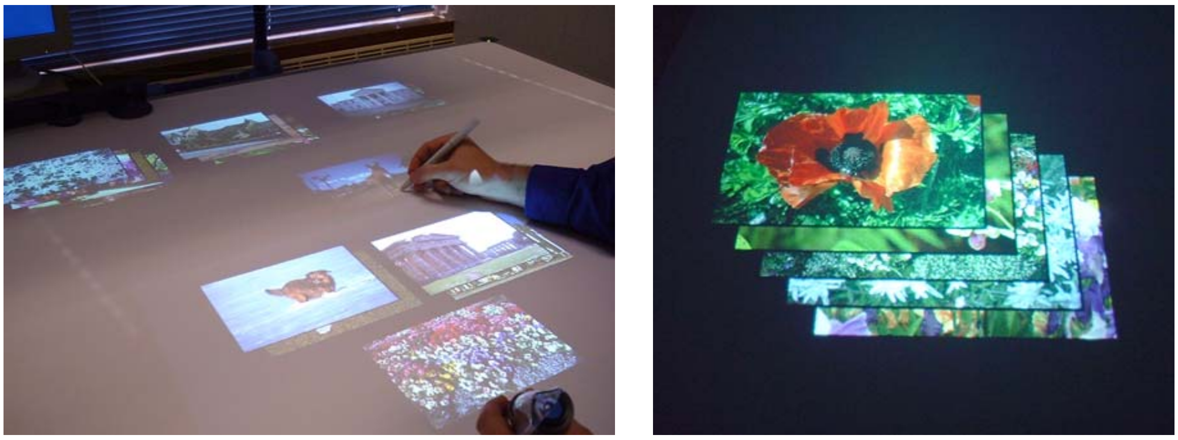


Figure 7.5: The second task the users were given was to sort images into piles: (left) the images started in random piles along the far edge of the desk and were rearranged into new piles; (right) the user made a pile for each type of image—flowers in this case.

Next the user was introduced to piles. He was instructed on the techniques shown in Figure 7.1—adding an item, re-ordering the pile, removing an item, and browsing—and given a chance to practice them. The second task was to take 15 images arranged randomly in three piles, and rearrange them into three new piles according to their categories of flowers, animals and buildings (Figure 7.5). There were five images in each category. This task was performed four times with a slightly different set of images each time, using the four pile directions shown in Figure 7.1(e) in a random order. The user was asked which pile direction he preferred, and was asked for comments on the piling system.

Finally the user was asked for comments about the digitizer pen, comments about the Mimio ultrasonic pen, and comments about the system generally. He was also asked if he thought it would be useful to have a visual cue to the extents of the fovea since there was none and I was interested in whether the sheets of virtual paper could be quickly brought to the fovea, even though the section of the desk that it occupied was not marked. The whole test took 30 to 40 minutes for each participant.

7.3.2 Results

All of the participants required only a few minutes of practice to be able to use the system to move and annotate the sheets of virtual paper. One remarked that it was easier to use than he had anticipated from the description of the system, and another said “using two pens was more intuitive than I expected”. Regarding the three pen cursors, one user was split between preference for no cursor and preference for the cross hair, which made the totals 5.5 for no cursor, 1.5 for the cross hair, and 0 for the trace. Comments about the trace included “I wasn’t sure if I was drawing or not” and “it would drive me mad”. It seems that in a single-user system with good calibration and low latency there is no need for feedback on present or past pen locations.

When asked about pile directions 4 participants did not express a preference, 1 said north-west, and 2 said north-east. One said that the item on top of the pile is the most important and so should be nearest to the user, which would account for north-west and north-east getting votes and south-west and south-east getting none. No effect of left-handedness was shown by the data. Generally pile direction was not considered to be especially important for the picture sorting task. Participants found it easy to add items to piles, re-order piles, and remove items from piles. The pile browsing mechanism was often initiated accidentally because the pen operates at up to 25mm from the digitizer surface—a pile would split unintentionally and items in it would move, causing some confusion. I added lift events to the system (Section 6.2.1), which signal the pen’s removal from the display surface, to alleviate this problem. When the user stands back to review the state of the desk, a pile that was being browsed will be sent a lift event that causes it to return to its default state. One of the digitizers listed in Appendix B generates *proximity* events when the pen is removed from the surface so these are used to trigger the lift events, but the other digitizer does not so a pause in the stream of events is used as the trigger. When the pen is on the surface the high accuracy of the digitizer and slight movement in the user’s hand cause events to be generated regularly—a period of 0.7 seconds with no events has proved to be a good indicator of the pen being removed from the digitizer surface.

As soon as visitors to the Computer Laboratory were introduced to the prototype interface it became apparent that the sensing of the digitizer buttons would have to be designed carefully. The digitizer pen has three buttons. In Figure 7.6 the button in the nib is labelled button 1, and those on the shaft are labelled button 2 and button 3. Writing must be performed using button 1 because this is the only way to detect when the user is pressing the pen against the surface. Operations that are invoked with buttons 2 and 3, such as erasing previously drawn marks and dragging sheets of virtual paper, must be insensitive to button 1 being pressed during the operation because users find it very difficult to use the pen with button 2 or 3 held down, without also, at least occasionally, pressing the nib button. I wrote the event handling system at the client so that if buttons are pressed simultaneously only one button press is actually reported to the server—the one from the button that was pressed first. This removes the problem of multiple buttons being pressed accidentally, although some users had to be taught to press the buttons on the shaft of the pen first when starting an erasing or dragging action.

The angle at which a user holds a pen to the display surface affects the ease with which the pen can be used. Both pens work best when they are held perpendicularly to the surface but this does not come naturally to users. The emitter of the ultrasonic pen is about 15mm from the its tip (Figure 7.7) so if the angle of the pen is varied its reported location changes, and varying the angle through 90 degrees can change the reported location by up to 15mm in each direction. If the ultrasonic pen is held at a constant angle the effect will be removed during calibration. In practice the variations in reported location were not a problem, probably because the ultrasonic pen is only used to drag sheets with the non-dominant hand which does not require high precision. The button in the nib of the digitizer pen is designed to press in as if the pen is perpendicular to the writing surface (Figure 7.7). Some users had difficulty using it because they held the pen obliquely then tried to apply a force perpendicular to the surface to engage the nib button rather than pushing along the length of the pen. After some instruction they modified their grip on the pen or changed the way they push the nib button to overcome this limitation of the hardware.

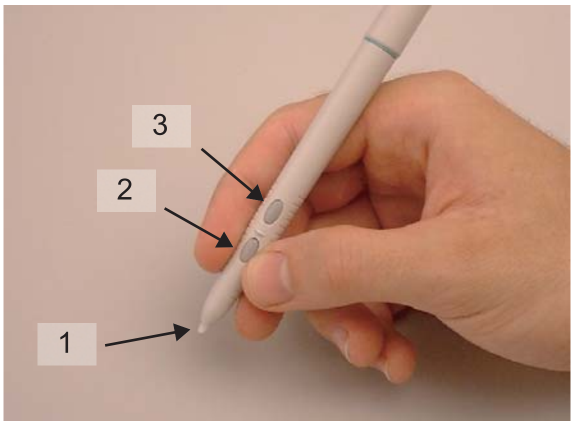


Figure 7.6: The digitizer pen for the dominant hand has three buttons: one in the nib, and two on the shaft. How combinations of buttons are interpreted by the software must be designed carefully.

The forms of the Mimio and digitizer pens are very different, which may have been a constant cue to the complementary affordances of the devices. No one confused the roles of the two pens by, for instance, trying to write with the Mimio pen. One participant simply noted that one pen seems large compared to the other one, while another said “the different shapes made me aware of the different functions”. To obtain data from the Mimio I used the software that came with it to make it emulate the mouse, then programmed the Escritoire client to process the mouse events that were subsequently generated. The problem I encountered with this approach is that if the pen is held in one position on the surface for over a second the Mimio driver generates a mouse click event rather than continuous mouse movement events. One test participant had to be taught to move the pen quickly rather than pressing it down then deciding where to move it—using the pen that way caused the sheet that the participant had intended to drag to be left behind. This problem would be fixed by gaining access to the raw data from the Mimio which is currently gathered from the device using a proprietary protocol.

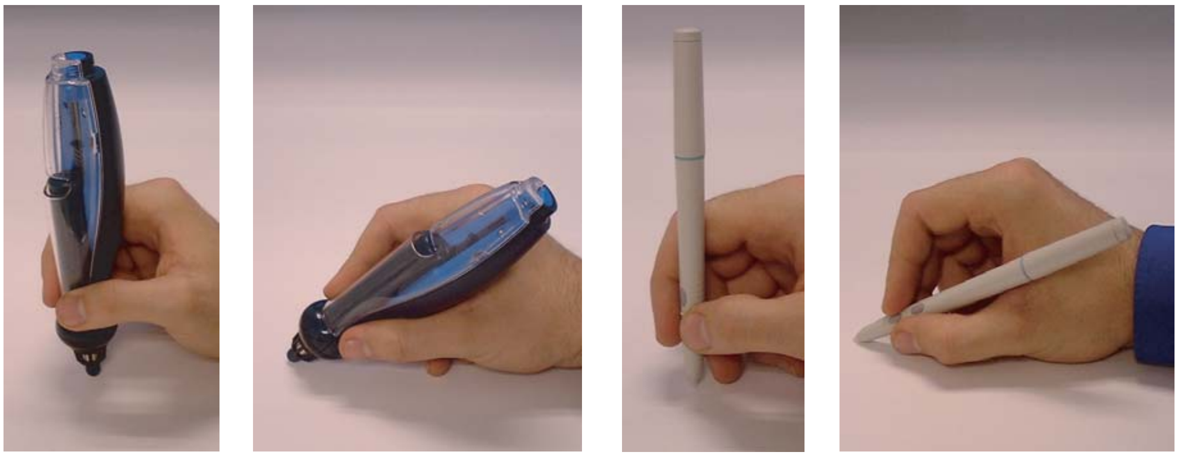


Figure 7.7: Holding the pens obliquely can cause problems. The Mimio pen reports variations in position of up to 15mm in each direction if it is held at very acute angles (left two images). The digitizer pen's nib button can be difficult to engage when it is held obliquely (right two images).

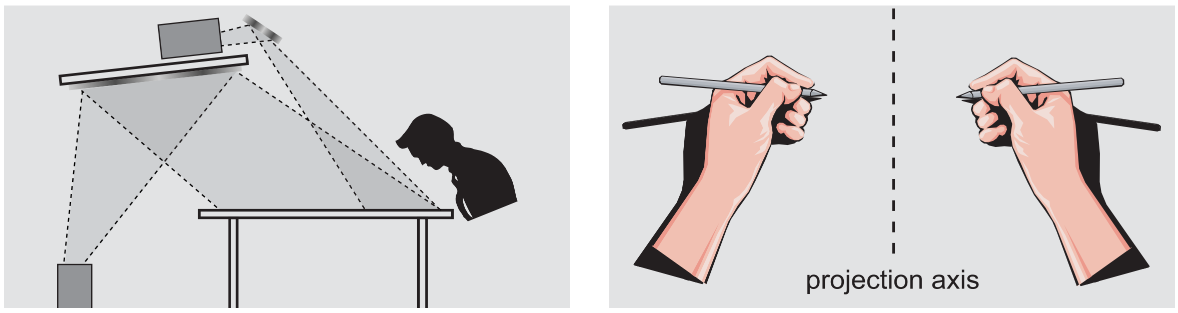


Figure 7.8: Occlusion of information is not a problem: (left) oblique projection allows the user to lean over the desk to get a closer look; (right) the fovea projector is mounted above the foveal region and in front of the user—when the user is working in the fovea the projected information near the pen tips is not occluded by the hands.

The digitizer provides accurate and timely events via Wintab [WT] but one participant did complain about the inactive area at the bottom of the surface. The Summagrid and GTCO Calcomp digitizers listed in Appendix B each have an active area that senses the pen, surrounded by a border of approximately 125mm that is inactive. The projectors that create the foveal display of the Escritoire can produce projected imagery across the whole surface including the border, but it can be misleading if they allow the sheets of virtual paper to protrude into areas where they cannot be manipulated using the digitizer pen. Confining the projection to the active area delimits the extents within which the digitizer pen can be used.

As with the DigitalDesk [Wel94, page 26] front projection was not a problem. No users complained about the images that were occasionally projected onto their arms, or about occlusion of information. I believe that occlusion was not a problem for three reasons. First, front-projection presentation systems force the user to be careful not to walk between the projector and screen, and this problem has also been present in projected interfaces for drawing [SHC$^+$95, page 15], but oblique projection from a location near the back of the desk allows the user to lean forward to get a better look without occluding information (Figure 7.8 (left)). Second, the fovea projector is mounted so that when the user's hands are working in the fovea the projector is in-between them, which means that the shadows fall away from the area between the hands (Figure 7.8 (right)). Third, people are accustomed to their hands shadowing light from above so they instinctively move them if the information underneath is shadowed.

Some ergonomic issues with the interface of the Escritoire became apparent during the tests. Two of the participants leant on the digitizer which caused it to move somewhat. The digitizer at Thales, which was used for these tests, does have a tendency to do this, and sturdier fittings such as those on the digitizer at the Computer Laboratory would have been preferable. The details of the two digitizers are given in Appendix B. An A0 digitizer has a large surface, and the digitizer used in the single-user tests was quite high above the chair that was provided. One participant said he liked this because he was big and could reach the whole thing, but another had to stand to reach the items at the back which was awkward. In configuring a system for a specific user, factors such as the chair height and digitizer position would be chosen to suit them—a standard pedestal allows a digitizer to be placed at a range of angles other than horizontal, and the calibration and warping system described in Chapter 4 would adapt the display system to permit this.

7.4 Collaborative tests

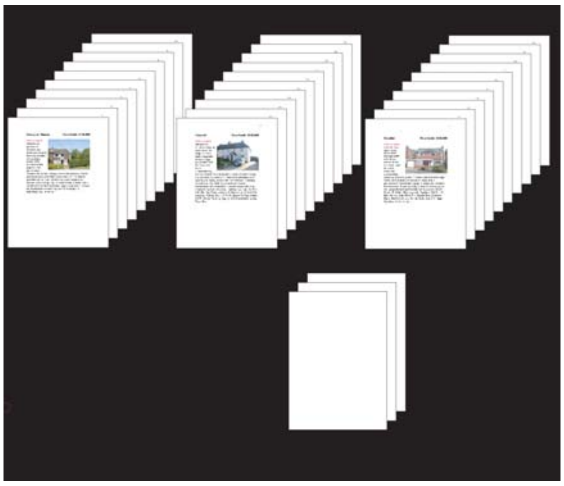
I augmented the video and audio channels of a traditional video conference with interaction between two Escritoire desks to show how a task space can complement a person space [AR03d]. Three pairs of participants performed a realistic task using the system. One person from each pair was at the Computer Laboratory in Cambridge and the other was at Thales Research & Technology in Reading, about 100 miles away.

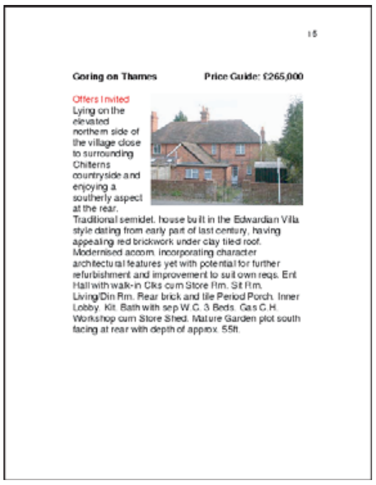
7.4.1 Method

The aim of the task was to pick, from a group of houses, the three of best value, and put them in order. The subject of house prices was bound to elicit discussion between the participants owing to it being second only the the weather as Britain's national topic of conversation. Initially each participant used the desk on his own. He was shown 30 houses and asked to look at them and use the blank sheets of virtual paper provided to make any notes that might prove useful for the later discussion (Figure 7.9). This stage took 20 to 30 minutes for each person.

For the collaborative part of the test, the server and one of the participants was at Thales, and the other participant was at Cambridge. The hardware of the client machines is listed in Appendix B and the server computer had the same specification as the Thales client computer. The server was linked to the remote client over the Internet which was accessed via a standard DSL connection delivering 256 kbps which was shared between the audio, video and desk. Each participant had an LCD monitor and a webcam behind his desk, and a videoconference link was created using Microsoft NetMeeting [NM] so the participants could see and talk to each other. The pair of participants performed the same task three times: given 10 of the original 30 houses they found the best value house, the second best, and the third best. They used a different pen cursor each time—no cursors, cross hairs, and traces—in a different order for each pair. Each group of 10 houses took took 20 to 30 minutes to discuss.

After completing the tasks the participants were asked which cursor type they preferred, and were asked for comments on the cursors. They were asked for a response to each of the following statements from 1:strongly agree to 5:strongly disagree.





15

Goring on Thames **Price Guide: £265,000**

Offers Invited
Lying on the elevated northern side of the village close to surrounding Chilterns countryside and enjoying a southerly aspect at the rear.
Traditional semidet. house built in the Edwardian Villa style dating from early part of last century, having appealing red brickwork under clay tiled roof. Modernised accom. incorporating character architectural features yet with potential for further refurbishment and improvement to suit own reqs. Ent Hall with walk-in Clks cum Store Rm. Sit Rm. Living/Din Rm. Rear brick and tile Period Porch. Inner Lobby. Kit. Bath with sep W.C. 3 Beds. Gas C.H. Workshop cum Store Shed. Mature Garden plot south facing at rear with depth of approx. 55ft.

Figure 7.9: The houses used in the collaborative task: (left) the 30 houses used in the task were examined by the users individually before they met to collaborate, and each user could make notes on the blank sheets provided; (right) each house was described on a single page which contained a textual description and a photograph.

- The audio channel was useful for the task
- The video channel was useful for the task
- The desk interaction was useful for the task
- The amount of desk space available aided the task

The participants were then asked whether the difference in resolution between the fovea and periphery was a problem, whether the difference in brightness between the fovea and periphery was a problem, whether latency in the desk interaction was a problem, and finally they were asked for general comments about the system.

7.4.2 Results

All of the participants understood the concept of interacting through the desk while conversing over a video conference link. After they had used the desk system on their own they did not need any extra training to use it for distributed collaboration. Figure 7.10 shows two participants interacting.

One user was split between preference for the cross hair cursor and the trace which made the totals 0 for no cursor, 0.5 for the cross hair, and 5.5 for the trace. This is almost the opposite of the single user case when the absence of a cursor was clearly preferred. One participant said that pointing to things, especially on the map he had drawn to help with the house evaluation task, was difficult when no cursors were present. Another said he would normally hate traces when working on his own, but they are useful to show someone else what you are doing. Traces proved to be useful in allowing a participant to direct the attention of the other, for example when pointing out details in a photograph.

The responses to the four statements are shown in Table 7.1 below. Participants clearly considered the audio and desk interaction essential for undertaking the task, but deemed the video channel to be relatively unimportant. The large amount of space available on the desk was considered an advantage.

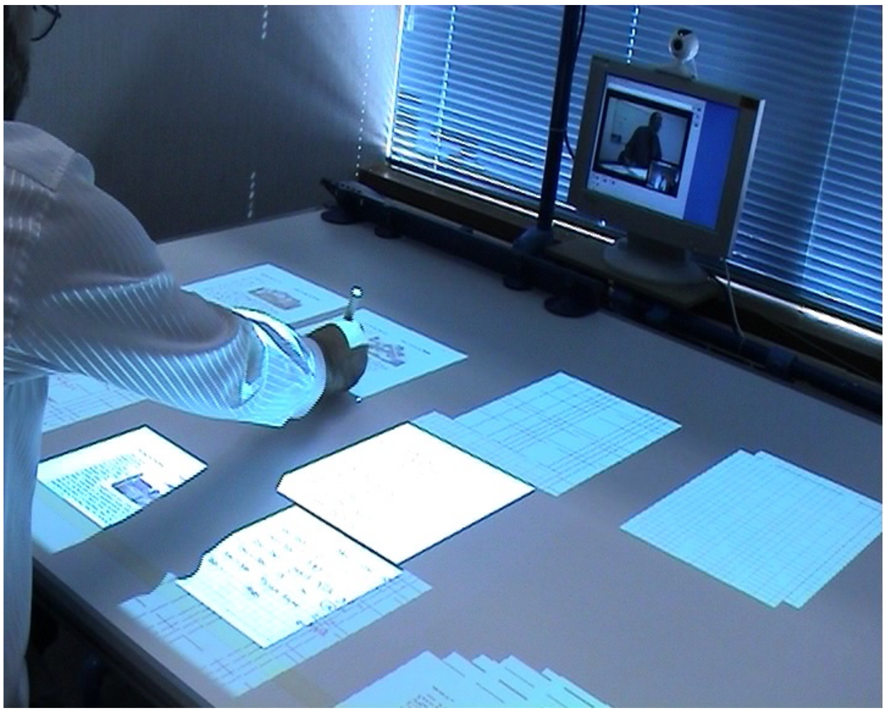


Figure 7.10: Sharing a task space. These two participants from the collaborative tests are in different cities, and are talking over the video link while interacting with the desk contents, which are available to both of them. The remote participant is visible on the screen in the top-right. The local participant is moving a sheet on the desk.

	strongly agree	agree	neither	disagree	strongly disagree
audio useful	6	0	0	0	0
video useful	0	2	2	0	2
desk useful	6	0	0	0	0
space helped	3	2	1	0	0

Table 7.1: Responses from the six participants of the collaborative tests to the statements listed in Section 7.4.1.

When asked whether the difference in resolution between fovea and periphery was a problem, 3 participants said no. One said not really, but on a real desk you would not always drag items in front of you to write on them; one said it is only a problem because you cannot read text that is outside the fovea; and one said that ideally the whole desk would be high resolution. Of course, the last point is true, but the reason for using a dual-resolution display is to balance the needs of the user against the cost, size, and complexity of the system.

The foveal regions of the two desks were in slightly different positions, which meant that sometimes one person would move a sheet in to read it, but then the other would then have to adjust its position so he could also read it. Three participants complained about this. The projectors for the two desk displays were not positioned precisely. Instead the top projector was positioned so that it covered a region at the bottom middle of the desk that was roughly the desired size, then the calibration procedure was used to choose a foveal region inside this. The problem of misaligned foveas could be solved by defining a fixed size and position for the fovea in advance, then positioning the top projector so that it covers that region without wasting too many pixels around the perimeter.

When asked about the difference in brightness between fovea and periphery, all participants said it was not a problem. One said he didn't really notice it, and another said it is useful because you can tell where the high and low resolution regions are. One said the the bright fovea region might be hard on the eyes although he hadn't found it to be a problem during the 1.5 hours for which he used it during the collaborative test. Five of the six participants did not notice any latency in the desk interaction, or said it wasn't a problem. One said it was only a problem when resolving actions between two people.

One participant said the ultrasonic pen for the non-dominant hand was not very useful and tended not to use it. Another used both pens with his right hand, switching between them as necessary. He said having many buttons on one pen was awkward, and it would be easier to have a cup of pens with different functions, which could be coloured to remind the user of the function of each one. The Intuos range of graphics tablets from Wacom [Wac] supports a series of different pens which have identification numbers that are sensed by the tablet so a different function can be assigned to each pen. The pens can be kept near the tablet and picked up and used as needed. Unfortunately the tablets are only available in sizes up to A3. One participant said he would have liked to have been able to move piles and items in piles with his left hand, leaving writing and erasing to his right hand, thus assigning to separate hands the tasks of arranging sheets and annotating sheets. Another said essentially the same thing—that he would like to be able to write with his dominant hand while browsing a pile with his non-dominant hand. The precise assignment of roles to hands and functions to pens and buttons could benefit from further testing and refinement, but it will be dependent on the current task and the exact interface features that are offered to users.

Conflicts between participants trying to move or annotate the same sheet were rare and not problematic. Two of the pairs used social protocol to avoid conflicts—for example one pair arranged that one of them would bring previously unseen houses into the fovea, while the other would move them out when they had been examined. Two participants commented that it would be nice to have a private workspace for making notes that were not visible to the other party. The division between public and private workspaces has been explored in many projects. For example: Courtyard [THY$^+$94] (Section 2.2.2) allowed items on a large shared screen to be viewed in detail on users' individual workstations; Jun Rekimoto's work[Rek98, Rek00] (Section 2.2.1) on wall and table displays allows material to be prepared on a laptop computer or PDA then moved to a public wall or table display; and Roomware [SPMT$^+$02] (Section 2.5.1) is an environment where information can be prepared on, and moved between, displays of various sizes embedded in chairs, tables, and walls. Another suggestion was that a magnifying glass could appear near the pen to make it easier to read text. This would be like a magic lens [BSP$^+$93] (Section 2.4.3) or the passive lens [UI97] which was an extra physical device used on the metaDESK. Its location was tracked, and the part of the display visible through its aperture was rendered differently (Section 2.4.3). Finally, one participant would have liked a facility to sort the piles of sheets of virtual paper describing houses according to attributes such as price and number of bedrooms. The Pile Metaphor was designed as an output format for an information retrieval system for documents [RM$^+$93b] and supported this kind of sorting based on rules entered explicitly by the user or automatic clustering of documents into piles based on a vector model.

7.5 Summary

People often create piles on their desks to save space and to avoid the premature categorization that conventional hierarchical computer filing systems impose. I have implemented piles for the items on the Escritoire's desk display, and have described a set of techniques for forming and browsing them with the pen in the dominant hand—the non-dominant hand is used to move a pile as a unit. Much of the interaction by collaborators in a task space consists of gestures rather than drawing or editing actions, so I have implemented pen traces to support awareness between collaborating users. I have described an efficient method for drawing traces because a shared space may often contain multiple traces that must be redrawn with every screen refresh.

First I tested the Escritoire with individual users. They could easily use the system after only a few minutes of training, they were not confused between the pens which had very different physical forms, and oblique projection from the rear of the desk meant that occlusion of the projected display was not a problem. I have described various issues and enhancements such as the use of proximity events to sense when the pen is taken off the digitizer surface. After the single-user tests I performed collaborative tests in which pairs of users communicated through a standard video conference, and also through the task space created by their two desks. The participants did not need any more instruction to use the desks for collaboration, almost all of them preferred the trace to the other cursor options, and they regarded the audio and desk channels as essential to their task but the video channel not so useful. This reinforces the view that a task space can be be much more useful than a person space for a task involving visual material.

Chapter 8

Conclusion

This chapter starts by outlining areas for future development and research based on the new developments and findings I have already described. These areas include changes to the hardware and calibration of the displays, new input and output devices, and various possibilities for the interface that would require further user testing. A summary then concludes this dissertation by reviewing the work I have undertaken and its major results, and by describing the general principles I wish to impart.

8.1 Future Directions

The building and testing of the Escritoire have suggested various directions for future development and research, ranging from basic additions to the hardware and software of the current implementation, to open-ended areas of study made possible by the novel interface. A mundane requirement is that the desk be sturdy enough for users to lean on, because they have a tendency to do this, especially when reading, as noted in Section 7.3.2, and this point should be added to the specification of a digitizer for a horizontal display. Rather than a standard-size digitizer, a scalable tracking surface that is made of tiles could be used to create an interactive surface of any size. Researchers at the Tangible Media Group at the MIT Media Lab are working on such a system [PIHP01] that tracks multiple cordless devices. Currently the tags that are tracked are 15 mm wide and the tracking is only accurate to 4 mm [PRI02], but with smaller tags that could fit inside pens, and higher accuracy, this type of scalable tracking technology could support bimanual input over a large area using only a single device. The hands could use identical pens, with the roles of the two being assigned by the software.

Section 4.2.1 describes how the transformation $\mathbf{H}_{\mathrm{pd}}$ from pen space to desk space is calculated. It is derived from a display rectangle for each projector that is positioned and scaled in an indirect manner using the keyboard. A better method would be to manipulate it directly with the pens as if it were a window in a conventional window system. This would follow the principle of manipulating items on the desk directly with pens, and would be quicker. The ultrasonic pen is calibrated using piecewise linear mapping (Section 5.1.3), which I have proved to be a good model for the desired function. Further work could investigate the use of Goshtasby's rational Gaussian surfaces [Gos99] which have the advantages that there is no distinction between points inside and outside the convex hull of the control points as there is with the piecewise linear mapping, and that the formulation is parameterized on the extent to which it creates a local, versus global, approximation.

Luminance and chrominance matching for multi-projector displays is an active area of research [MS03]. The luminance difference between the fovea and periphery of the Escritoire's display is actually useful (Section 7.4.2) and, in any case, achieving the same brightness for the two regions would be undesirable because most of the intensity of the foveal projector would have to be wasted to bring it down to the level of the periphery. The chrominance difference between the regions has not been a problem in practice but chrominance matching could be included to increase the continuity between regions at the expense of adding a camera to the system. The optical properties of the digitizer surface would have to be considered—the digitizer used at the Computer Laboratory (Appendix B) exhibits non-Lambertian reflection which is not ideal and would require the camera to be mounted close to the line of sight of the user.

A planar homography is a good approximation to the distortion experienced by the image that is projected onto the desk surface as shown by the results in Section 5.1.4, but nonlinearities do occur near the corners of the periphery because the projector has a lens and a finite aperture. Just as a piecewise linear mapping was used for the ultrasonic pen, a piecewise projective mapping could be used for the graphics warping by dividing the plane into quadrilaterals, so that the warp could approximate non-projective mappings such as radial and tangential lens distortion. A grid of quadrilaterals could be positioned to coincide with a grid printed on a large piece of paper by dragging the intersection points with the digitizer pen, or a camera and a specialized computer vision algorithm could be used to position the points automatically.

Video cards now often have 256 MB of memory. At 72 dots per inch and 16 bits per pixel that is enough to store over 200 A4 sheets. A different texture could be used for each sheet on the desk, then each one could be drawn as a quadrilateral made of two triangles and would have a separate planar homography transformation that would translate it from the origin to its desired location on the desk before the transformation is applied to warp the sheet to account for oblique projection. This approach would allow any planar homography to be used to transform each sheet so, without any extra computational cost, rotations could be explored as in Beaudoin-Lafon's work [BL01], and scaling could be used to save space as in Zoomscapes [GSW01]. It would also avoid moving large amounts of data across the computer's bus to update the displays when a sheet is moved, although care would still have to be taken to optimize updates to the textures.

Further user testing could provide insights regarding the best assignment of roles to the two hands. Two of the users in the collaborative tests said they would have liked to have been able to move and browse piles with the left hand, while annotating with the right hand (Section 7.4.2). This follows the principles presented by Guiard [Gui87] that the left, non-dominant, hand operates first and performs larger, coarser movements. If the interface of the Escritoire was augmented to provide more features for general tasks there would have to be some way to choose modes or tools. Marking menus [CHWS88] or flow menus [GW00] could be used to keep control at the location of the pen, rather than using conventional buttons and menus. Toolglasses and magic lenses [BSP$^+$93] could also be added to the interface to make use of both hands. Extra functions could be accessed by instead using alternative input devices. Buxton advocates the use of specialized tools for specific tasks because they give better performance [FB97]. A cup of pens would allow functions to be accessed easily, and the user could hold multiple pens in one hand or drop them on the desk as one does when using multiple coloured pens and pencils. The Wacom [Wac] Intuos range of tablets support a range of different pens although they are not available in a size large enough to fill a desk. A nice feature would be having an active tip on both ends of a cordless pen, so, for instance, one end of a pen could used for writing and the other for erasing. This is available on Wacom Graphire tablets. An alternative to an interface with two pens is one with a pen for the dominant hand and a puck for the non-dominant hand. The puck might tend to obscure information on the display, but multiple modifier buttons on it could be used while the dominant hand simultaneously performs detailed work, which would be similar to the way in which a keyboard and mouse can be used together.

The difference in the position and size of the foveal regions of the two desks was an issue during collaboration. Occasionally one user would arrange some items in the fovea of his display, then the other user would have to adjust the positions so she could also see the detail on those items. Because I have chosen to faithfully render items on the desk at their real size, ensuring that the foveal regions are coincident would require the position of the fovea on the desk to be defined in advance. The peripheral projector could be calibrated, then it could highlight the predefined area that should be covered by the fovea. The foveal projector could then be manually adjusted so it covers that area without wasting too many pixels. Two of the six participants in the collaborative tests said they would have liked to have had a private area that was not accessible to the other participant. Private workspaces could be added to the system, and the projected wall display could also be used for storing personal files that could be brought to the shared desk surface at any time during the collaborative session.

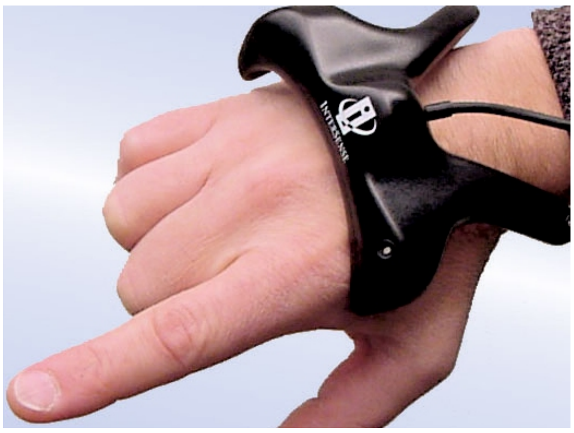


Figure 8.1: The Intersense MiniTrax ultrasonic hand tracker [MT]. A device like this would avoid the problems that a magnetic tracker has with metal objects, and would leave the hand free to hold a pen.

I believe there is great potential for the wall display as a repository for background tasks because it allows them to be identified at a glance and brought quickly to the desk. A large wall display could even be used in this way with a normal monitor as the high-resolution area, thus creating a system like Kimura [MMV$^+$01] which is described in Section 2.2.2. The wall display can act like a bookshelf, holding regularly used items. The Polhemus magnetic tracker I have used to remotely control the wall display has the drawback that it cannot be used near metal, which is difficult to avoid in a normal office environment. An ultrasonic device, such as the Intersense MiniTrax (Figure 8.1) described in Section 5.2.2, would avoid this problem, and the receiver could be mounted on the shelf holding the foveal projector which would ensure that line of sight between emitter and receiver is maintained. If a camera for each display surface was added to the system, a laser pointer could be used to move items on the desk and wall displays, which would cut down the degrees of freedom of the input from six to just the two that are necessary. An alternative method for calibrating the graphics warping for the wall display would be needed. Sukthankar *et al.* [SSM01] have used a laser pointer to control presentations on a projected display from a distance, by using a standard laser pointer and converting a dwell in a particular location to a click event. I think it would be much better to add a radio button to the laser pointer as Guimbretiére has done [Gui02, page 51] to an ultrasonic whiteboard pen for controlling the Stanford Interactive Mural, because then the start and end times for a gesture would be precisely measured by the system. The use of a standard laser pointer with an extra button is supported by recent empirical results. Myers *et al.* [MBN+02] found that users could not position a laser pointer beam precisely when turning it on or off, and that holding the laser pointer like a pen or mounting it on a gun was not helpful, and that it was better to use the standard grip in the fingers. Cavens *et al.* [CVFM02] have shown that a pointer with a radio button is as good as a mouse for selecting targets on a large projected display. In a system like Kimura a laser pointer would have to be pointed at a conventional monitor which I believe would be unsatisfactory as an interface and would cause problems for sensing with a camera, but a laser pointer could be used with multiple front-projected displays, and it would have the advantages of being small and wireless like the digitizer pen.

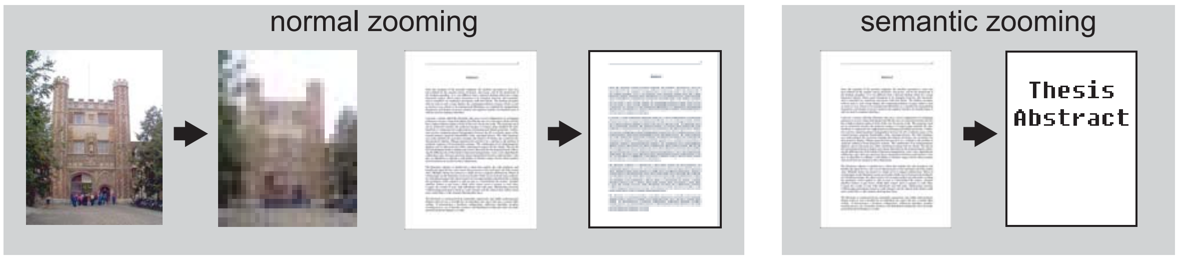


Figure 8.2: In normal zooming (left) graphical content is simply displayed at a lower resolution when the user zooms out. This works well for images but not so well for text because the words quickly become unreadable. In semantic zooming (right) the text is replaced by a more concise version that is still readable.

When an image is moved from the fovea to the periphery it is displayed in a lower resolution, which reveals the general form of the image but not the fine detail. The same is true of textual documents but then the text becomes unreadable in the periphery. When a document is moved to the periphery, rather than displaying the original text at a lower resolution, semantic zooming could be used to display a version of the document with less text (Figure 8.2). This would require metadata for documents that could be added manually, or possibly automatically generated from the source file.

Instead of using a wall display, more space for arranging documents could be provided by relaxing the rule that items on the desk are always displayed life-sized and using a visualization technique like the Document Lens (Section 2.1.2). Because the contents of the desk are drawn as a texture-mapped quadrilateral virtually no extra computational cost would result from using five quadrilaterals that fill the same space (Figure 8.3). More than two projectors could be used, but they should not be arranged in a series with each display containing a smaller one, because all but the smallest projected display would waste many of its pixels by leaving them black because they are overlapped by a higher-resolution projected region. Instead, more projectors could be used to create a larger fovea or multiple foveal regions. A movable fovea could be implemented using a device like the Everywhere Display (Section 2.3.2) to shift the foveal region between predefined positions. No system currently exists to create a projected display that can operate continuously at it moves so that it could follow the tip of the pen. The fundamental requirements of such a system would be mechanical ones: the ability to quickly and accurately control the pan-tilt mirror and the focus of the projector.

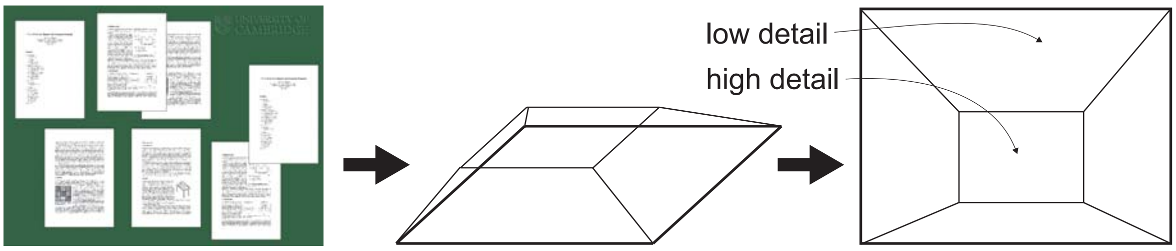


Figure 8.3: A visualization technique like the Document Lens could be used on the desk. The image of the desk contents would be texture-mapped onto a five-sided shape at virtually no extra cost.

Electronic documents are added to the Escritoire system by printing to PDF as if printing to paper, then loading the PDF file. Pages from physical papers and books could be easily added if a scanner was placed next to the desk display, and if the use of a scanner is considered to disruptive to the work flow, a tangible interface like FlowScan (Section 2.4.3) could be incorporated. The ability to sort items in a pile based on metadata would be useful: for instance, in the task in which participants collaborate to choose houses (Section 7.4) it would be useful to sort the houses into price order. This facility would be close to Apple's vision of piles as an interface to an information retrieval system [RM$^+$93b]. Piles could also be sorted by information added to the items with the pen—for instance, the user could write numbers on the edges of some sheets, then they would be sorted into ascending order in the pile.

8.2 Summary

In manual tasks, humans make use of large workspaces to allow them to access many items, to exploit their ability to glance around and use peripheral vision, and to use the full reach of both hands and their kinæsthetic sense to locate items quickly. People use desk organization to remind themselves of tasks that must be done, and to loosely categorize items into piles; an approach that complements more formal, long-term filing systems. The conventional computer interface uses a keyboard, a mouse, and a vertical screen with a diagonal of around 17 inches to display a desktop metaphor that does not support these natural modes of working. I have created a system called the Escritoire that does support these behaviours, and that demonstrates some issues and solutions in physical configuration, use of video hardware, calibration, interface design, and use for remote collaboration. Unlike papers strewn over a real desk the virtual mess on the desk display disappears at the touch of a button. The Escritoire is a projected desk display with bimanual input that is constructed from standard hardware components. It is quickly calibrated and new users have found it easy to use for individual and collaborative work.

Two projectors overlap to create a *foveal display* that fills the desk with a low-resolution periphery, and also provides a high-resolution fovea in front of the user. This combination of projectors creates a horizontal display that is qualitatively different from a conventional monitor screen because it is large enough to contain many documents, images, and programs, and allows the user to glance at them quickly and to bring any one to the fovea, which has the resolution required to display pages from an A4 document at life size so that they can be read and annotated. The desk itself is a large digitizer, which I have combined with an ultrasonic pen to allow simultaneous input from both hands over the entire area of the desk, as was originally desired. The direct pointing method of the pens is easy to learn, much more suitable for the large display than an indirect device like a mouse, and supports the kinæsthetic sense that allows users to locate items without looking directly at them. Users easily used two pens at once—the simple operation of the ultrasonic pen for the non-dominant hand meant than they could easily use it to drag items around the desk. The computers, video cards, projectors, and pen devices are standard components, so the cost of the complete set of hardware fulfils the objective of being feasible for a personal workstation, especially given the falling cost of projectors which are the most expensive part.

Rendering and display technology will continue to increase in power and decrease in cost, but, as always, users will be limited by their current hardware. In discussing variable-resolution rendering techniques for still images and video streams that take account of the direction in which the user is looking, Baudisch *et al.* [BDDG03] state that 'Techniques that attempt to match the characteristics of computer displays to the characteristics of human vision, namely its distinction between foveal vision and peripheral vision, will try to make better use of limited rendering resources by tailoring display content to the affordances of human vision. They suggest that such displays will be an enduring factor in the design of interactive computer systems.' The Escritoire's foveal display is the hardware equivalent of this type of approach, and it too allows the user to stay ahead of the hardware performance available from projectors.

The whole front-projection system requires little extra floor space over that of the desk, it fits below a typical office ceiling, and does not require complex mounts for the projectors or precise mechanical positioning—further attributes that make it suitable as personal display. The projected imagery is warped in real time using commodity video hardware to compensate for rough projector positioning, and my implementation of this warping system with DirectX and commodity graphics cards shows that it can easily deliver interactive frame rates of 30 FPS or more, and that a two-projector display can be calibrated in around 90 seconds. Projector technology is continually improving. High-end mass-market projectors currently have 1024×768 resolution, but the 1600×1200 projectors that should be become economical during the next few years would give the fovea of the Escritoire's display similar visual fidelity to a typical LCD monitor. Because of the way the projectors are mounted under the desk and on a shelf above it, motorized remote-controlled zoom and focus would increase the ease with which they are set up, and if future work used a pan-tilt mirror to make the fovea move, the focus and zoom would have to be controllable from the computer via a cable so that they could be adjusted continually as the fovea moves.

In Section 4.1.2 I defined 2D co-ordinate spaces for the images that are displayed from the projectors, the framebuffers of the projectors, the desk, and the pen devices, and I explained the planar homographies between those spaces which will need to be considered in the construction of any similar projection system. Calibration consists of calculating the homographies. I obtain the homography from pen to framebuffer by selecting a series of projected points on the digitizer—I have used nine points—then applying a closed-form least-squares solution. I manually obtain the homography from pen to desk by choosing the size, position, and orientation of the display rectangle on the desk surface. I have experimentally justified the use of planar homographies by demonstrating that they closely match the distortion experienced by the projected images. In Section 4.2.2 I outlined procedures for performing the projective warp with 3D video hardware using the two main graphics APIs Direct3D and OpenGL. Commodity video cards can easily perform the warping in real time, even for large images, using their perspective texture mapping facilities, but updating the texture rapidly is more of a challenge, and I have therefore given an algorithm in Section 4.2.3 to optimize this process by selectively updating the display using update regions. In addition to the algorithm to calibrate the projectors and digitizer pen using planar homographies, I have also given methods to calibrate less accurate 2D devices, and to calibrate 3D devices. I produced a variant of Piecewise Linear Interpolation which was originally used for image registration but can be used to calibrate a 2D input device, like the ultrasonic pen, without any prior knowledge of the mapping function required—control points can be added until the accuracy is high enough. I have also given a method to calibrate a 3D tracking device to a projected wall display which, rather than requiring the user to touch the surface, allows him to aim at projected targets from a distance so a display can be created on any free patch of wall or ceiling.

I have implemented sheets of virtual paper to display PDF documents, images, and VNC clients. The PDF and image sheets allow files in standard formats to be manipulated on the desk, and any annotations made on them are visible afterwards using standard viewing software. The VNC clients support conventional desktop metaphor applications, so they allow programs like web browsers and command line windows to be placed on the desk. The sheets can be put in piles to save space, and to group, order, and browse them. Test participants were able to use the Escritoire's interface after only a few minutes of practice—it is easy to forget how long it takes to be able to confidently use the keyboard and mouse, but the time scale will be measured in days or weeks rather than minutes. The ultrasonic pen for the non-dominant hand is chunky and easy to grab, and its operation is simple: just press down, move, and release. Participants found it easy to use in the non-dominant hand. Participants also generally found the digitizer pen easy to use after some modifications were made to the way functions are assigned to buttons, as described in Section 7.3.2. Most participants preferred to have no cursor showing the position of the pen, presumably because if the calibration is accurate the sensed location of the pen is obvious and a cursor just obscures the information underneath. The use of front projection rather than rear projection has not been a problem, and the locations of the projectors mean that the usual problem of occlusion does not occur—the user can lean forward to look at the display without occluding the projection, as shown in Figure 7.8 on page 112. The difference in resolution between the fovea and periphery of the display was not a serious problem for participants, and the difference in intensity is actually an advantage because it delineates the foveal region. In general, users that had no prior experience of the system could use it after only a few minutes of training for arranging and annotating documents and images as they would on a desk with physical sheets of paper.

The client-server architecture of the Escritoire separates the graphics processing and handling of input devices on the client, from the storage of, and event-driven interaction with, the sheets of virtual paper on the server. The use of a thin client that does not store any state means that the client can be restarted at any time. Use of the desk involves bursts of messages when the user is performing an action, with long gaps in-between when she is thinking, so I created a protocol, which is described in Section 6.1.3, that saves network capacity by switching between client-pull and server-push schemes for the messages from server to client. During periods of activity client-pull is used so the transmission is limited by the speed at which the client can process the incoming data, and events are buffered at the server and coalesced whenever possible. During periods of inactivity server-push is used, so the server sends out a new message as soon as it is received.

A video channel and audio channel, and collaboration via two Escritoire desks, were transmitted over a standard DSL Internet connection of the type available to most UK homes and businesses. The connection could easily support the three channels during a collaborative task, but the system would benefit from reducing, compressing, or progressively transmitting the bitmap data for the sheets that is sent at the start of a session, and a quality of service guarantee across the network would avoid the occasional breaks in the interaction because of high latency or packet loss. Test participants needed no extra training to use the Escritoire for collaborative work—they just started speaking to the other person and jointly working on the items on the desk. The pen trace was a useful addition to the interface for collaborative work: Of the 6 participants, 5.5 preferred it over the other options. This is partly because it is useful for making deictic references and gestures, but also because it supports peripheral awareness of the actions of other users, like when a pilot and copilot of a plane know what each other is doing even though they are not attending directly to each other's actions [Nor93]. Participants found the audio and desk channels useful, but the video not so useful. This reinforces my original assertion that in tasks based around visual or textual material, a task space is more useful for collaboration than a person space. When they use the shared desk as their focus of attention, collaborators are presented with cognitive artifacts that supplement their memory of the information that is available and the progress they have made.

There is currently much interest in miniature interfaces for mobile devices and PDAs, which is fuelled by advances in technologies such as small screens, batteries, and ubiquitous networks, but I believe that in fixed locations such as offices, where space and mobility are not limiting factors, large-format interfaces will become popular. The prevalence of multi-monitor systems indicates users' appetites for screen space. In a ubiquitous computing environment the small tab-sized devices and the large board-sized devices can be combined to complement each other's strengths [MT02, PPL$^+$03]. The large-format displays can be created from projectors, and the Escritoire is an example of such a display with a lower price and requiring less space than existing multi-projector display walls for visualizations and presentations. The desktop metaphor does not translate well to such large displays, but the ease of direct manipulation with pens makes it a good way to control a desk-sized display. The Escritoire is constructed from standard components and exploits users' existing manual skills to form a personal projected display for performing the everyday tasks for which people traditionally use their desks.

The world will become populated by a multitude of small mobile devices, but homes and offices where people spend much of their time offer the ability to interact with information on a larger scale, through stationary large-format display and interaction devices. These devices will create rich graphical workspaces that exploit the user's peripheral vision to make large amounts of information available without being intrusive. In 1946 John von Neumann wrote 'We are ... forced to recognize the possibility of constructing a hierarchy of memories, each of which has greater capacity than the preceding but which is less quickly accessible.' This concept should be extended to the user interface to construct a hierarchy of interactive surfaces, each of which has greater capacity than the preceding and is more coarsely rendered and controlled.

Glossary

A0–A4 Standard paper sizes in millimetres: A0 1189×841, A1 841×594, A2 594×420, A3 420×247, A4 297×210.

CAD Computer Aided Design.

column-major The method of storing the elements of an array that stores the first column, followed by the next column, and so on. See also row-major.

DLP Digital Light Processing. Invented by Texas Instruments, this system uses an array of tiny mirrors on a chip with conventional RAM address circuitry. The mirrors are tilted individually, and together they create the image emitted by a digital projector based on this technology.

double-buffering A process in which two framebuffers are used to smoothly move from one frame of graphical output to the next. The primary framebuffer is displayed on an output device such a projector, while the secondary framebuffer is prepared with the contents of the next frame. When it is ready the two framebuffers swap places, and the process continues.

dpi Dots per inch. A measure of the resolution of a display device.

DSL Digital Subscriber Line: a technology that allows digital information to be sent over standard copper telephone lines at speeds of around 512 kbps downstream and 256 kbps upstream, and is available to most UK homes and businesses.

GIS Geographic Information Systems.

glyph A visual marker that is placed on an object so that it can be identified by a computer vision system.

GUI Graphical User Interface.

homography Homography and collineation are alternative names for the projective transformation.

kbps Short for kilobits per second, a measure of data transfer speed. For example, 256 kbps is approximately 31 kilobytes per second.

LCD Liquid Crystal Display. LCD panels are used in digital projectors. Three panels provide the red, green and blue parts of the image.

MIP mapping A method of texture mapping where many versions of the texture are stored, from full resolution down to a single pixel, each version having half the width and half the height of the previous one. When a texture-mapped surface is drawn, the appropriate version of the texture is used so that the mapping from screen pixels to texels is as close to one-to-one as possible.

OCR Optical Character Recognition.

oblique projection The situation where the projector is not pointing perpendicularly at the surface, put makes some other angle with it. This causes the projected image to be distorted.

PDA Personal Digital Assistant. A hand-held computer that usually has wireless networking capability and is controlled by a stylus.

Prolog A programming language that is good for rapid prototyping and for problems with which logic is intimately involved.

row-major The method of storing the elements of an array that stores the first row, followed by the next row, and so on. This is how arrays in C are arranged in memory. See also column-major.

similarity transformation Similarity transformations, which are examples of conformal mappings, are rigid body transformations with an extra parameter for uniform scaling. The similarity transformation is a special case of the projective transformation.

SVGA A device with SVGA resolution has 800×600 pixels.

systematic error Persistent error in measurement whose form does not change over time, as opposed to random error.

texel Locations in a texture bitmap image are measured in texture elements known as 'texels'.

VGA A device with VGA resolution has 640×480 pixels.

Voronoi diagram Given n control points, a Voronoi diagram is a partition of the plane into n regions, where the locations in each region are closer to the corresponding control point than to any other control point.

widget A reusable component of a graphical user interface like a button or scroll bar. Widgets are usually grouped in a library which has a consistent look and feel.

XGA A device with XGA resolution has 1024×768 pixels.

References

[AAH97] Toshifumi Arai, Dietmar Aust, and Scott E. Hudson. PaperLink: A Technique for Hyperlinking from Real Paper to Electronic Content. In *Proceedings of CHI 97*, pages 327–334, 1997.

[ABM$^+$97] Maneesh Agrawala, Andrew C. Beers, Ian McDowall, Bernd Frhlich, Mark Bolas, and Pat Hanrahan. The Two-User Responsive Workbench: Support for Collaboration Through Individual Views of a Shared Space. In *Proceedings of Siggraph 97*, pages 327–332, 1997.

[AD01] Christine J. Alvarado and Randall Davis. Preserving the Freedom of Paper in a Computer-based Sketch Tool. In *Proceedings of HCI International 2001*, pages 687–691, 2001.

[Ado] Adobe. `<http://www.adobe.com>` (accessed 4 September 2003).

[AFSR03] Mark Ashdown, Matthew Flagg, Rahul Sukthankar, and James M. Rehg. A Flexible Projector-Camera System for Multi-Planar Displays. Technical Report IRP-TR-03-14, Intel Research, 2003.

[Ali02] Dzmitry Aliakseyeu. Direct Manipulation Interface for Architectural Design Tools. In *Extended Abstracts of CHI 2002*, pages 536–537, 2002.

[AMK95] Toshifumi Arai, Kimiyoshi Machii, and Soshiro Kuzunuki. Retrieving Electronic Documents with Real-World Objects on InteractiveDESK. In *Proceedings of UIST 95*, pages 37–38, 1995.

[AMR00] Yuji Ayatsuka, Nobuyuki Matsushita, and Jun Rekimoto. HyperPalette: A Hybrid Computing Environment for Small Computing Devices. In *Extended Abstracts of CHI 2000*, pages 133–134, 2000.

[An] Anoto. `<http://www.anoto.com>` (accessed 16 December 2003).

[AR01] Mark Ashdown and Peter Robinson. The Writing's on the Wall: Large, Remotely Controlled Displays. In *Proceedings of the First European Conference on Computer-Supported Collaborative Learning (Euro-CSCL 2001)*, pages 83–88, 2001.

[AR02] Mark Ashdown and Peter Robinson. The Escritoire: A Personal Projected Display for Interacting With Documents. Technical Report UCAM-CL-TR-538, University of Cambridge Computer Laboratory, 2002.

[AR03a] Mark Ashdown and Peter Robinson. A Life-Sized Desk Display for Peripheral Awareness and Remote Collaboration, 2003. First Research Workshop on Augmented Virtual Reality (AVIR 2003).

[AR03b] Mark Ashdown and Peter Robinson. Experiences Implementing and Using Personal Projected Displays. In *IEEE International Workshop on Projector-Camera Systems (Procams 2003)*, 2003.

[AR03c] Mark Ashdown and Peter Robinson. The Escritoire: A Personal Projected Display. In *11th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG 2003)*, pages 33–40, 2003.

[AR03d] Mark Ashdown and Peter Robinson. The Escritoire: Remote Collaboration in a Task Space. In *Proceedings of the ACM SIGMM Workshop on Experiential Telepresence (ETP 2003)*, pages 73–75, 2003.

[AS03] Mark Ashdown and Rahul Sukthankar. Robust Calibration of Camera-Projector System for Multi-Planar Displays. Technical Report HPL-2003-24, Hewlett Packard Cambridge Research Laboratory, 2003.

[ASM$^+$00] Anne H. Anderson, Lucy Smallwood, Rory MacDonald, Jim Mullin, and AnneMarie Fleming. Video Data and Video Links in Mediated Communication: What Do Users Value? *International Journal of Human-Computer Studies*, 52(1):165–187, 2000.

[AWT] Java Foundation Classes: the AWT and Swing libraries. `<http://java.sun.com/products/jfc>` (accessed 4 September 2003).

[BBT97a] Marla J. Baker, Lauren J. Bricker, and Steven L. Tanimoto. Cooperative Interaction Techniques for Graphical Objects in a Collaborative Activity. Technical Report 97-04-03, Department of Computer Science and Engineering, University of Washington, 1997.

[BBT97b] Lauren J. Bricker, Marla J. Baker, and Steven L. Tanimoto. Support for Cooperatively Controlled Objects in Multimedia Applications. In *Extended Abstracts of CHI 97*, pages 313–314, 1997.

[BCR03] Patrick Baudisch, Edward Cutrell, and George Robertson. High-Density Cursor: A Visualization Technique that Helps Users Keep Track of Fast-Moving Mouse Cursors. In *Proceedings of Ninth IFIP International Conference on Human-Computer Interaction (INTERACT 2003)*, 2003.

[BDDG03] Patrick Baudisch, Doug DeCarlo, Andrew T. Duchowski, and Wilson S. Geisler. Focusing on the Essential: Considering Attention in Display Design. *Communications of the ACM*, 46(3):60–66, 2003.

[Ber93] Philippe Bertrand. A Server-Less Window System for Multi-Tasking, Multi-Processor Systems. Master's thesis, University of Waterloo, Canada, 1993.

[BF91] Eric A. Bier and Steve Freeman. MMM: A User Interface Architecture for Shared Editors on a Single Screen. In *Proceedings of UIST 91*, pages 79–86, 1991.

[BG02] Patrick Baudisch and Nathan Good. Focus Plus Context Screens: Visual Context and Immersion on the Desktop, 2002. Siggraph 2002 demo.

[BGBS02] Patrick Baudisch, Nathaniel Good, Victoria Bellotti, and Pamela Schraedley. Keeping Things in Context: A Comparative Evaluation of Focus Plus Context Screens, Overviews and Zooming. In *Proceedings of CHI 2002*, pages 259–266, 2002.

[BGR$^+$98] Steve Benford, Chris Greenhalgh, Gail Reynard, Chris Brown, and Boriana Koleva. Understanding and Constructing Shared Spaces with Mixed-Reality Boundaries. *ACM Transactions on Computer-Human Interaction*, 5(3):185–223, 1998.

[BGS01] Patrick Baudisch, Nathaniel Good, and Paul Stewart. Focus Plus Context Screens: Combining Display Technology with Visualization Techniques. In *Proceedings of UIST 2001*, pages 31–40, 2001.

[BH94] Ben B. Bederson and James D. Hollan. Pad++: A Zooming Graphics Interface for Exploring Alternative Interface Physics. In *Proceedings of UIST 94*, pages 17–26, 1994.

[BJM00] Alan F. Blackwell, Anthony R. Jansen, and Kim Marriott. *Theory and Application of Diagrams, Lecture Notes in Artificial Intelligence (LNAI) 1889*, chapter on Restricted Focus Viewer: A Tool for Tracking Visual Attention, pages 162–177. Springer Verlag, 2000.

[BKP01] Mark Billinhurst, Hirokazu Kato, and Ivan Poupyrev. The MagicBook: A Transitional AR Interface. *Computers and Graphics*, 25(5):745–753, 2001.

[BL01] Michel Beaudouin-Lafon. Novel Interaction Techniques for Overlapping Windows. In *Proceedings of UIST 2001*, pages 153–154, 2001.

[BLMA$^+$01] Michel Beaudouin-Lafon, Wendy E. Mackay, Peter Andersen, Paul Janecek, Mads Jensen, Michael Lassen, Kasper Lund, Kjeld Mortensen, Stephanie Munk, Katrine Ravn, Anne Ratzer, Søren Christensen, and Kurt Jensen. CPN/Tools: Revisiting the Desktop Metaphor with Post-WIMP Interaction Techniques. In *Extended Abstracts of CHI 2001*, pages 11–12, 2001.

[Bly88] Sara A. Bly. A Use of Drawing Surfaces in Different Collaborative Settings. In *Proceedings of CSCW 88*, pages 250–256, 1988.

[BM86] William Buxton and Brad A. Myers. A Study in Two-handed Input. In *Proceedings of CHI 86*, pages 321–326, 1986.

[Bol80] Richard A. Bolt. “Put-That-There”: Voice and Gesture at the Graphics Interface. *Computer Graphics*, 14(3):262–270, 1980. Proceedings of Siggraph 80.

[Bol81] Richard A. Bolt. Gaze-Orchestrated Dynamic Windows. *Computer Graphics*, 15(3):109–119, 1981. Proceedings of Siggraph 81.

[Bol84] Richard A. Bolt. *The Human Interface*. Lifetime Learning Publications, 1984.

[Bri98] Lauren J. Bricker. *Collaboratively Controlled Objects in Support of Collaboration*. PhD thesis, University of Washington, Seattle, 1998.

[Bro92] Lisa Gottesfield Brown. A Survey of Image Registration Techniques. *ACM Computing Surveys*, 24(4):325–376, 1992.

[BSP$^+$93] Eric A. Bier, Maureen C. Stone, Ken Pier, William Buxton, and Tony D. DeRose. Toolglass and Magic Lenses: The See-Through Interface. In *Proceedings of Siggraph 93*, pages 73–80, 1993.

[Bus45] Vannevar Bush. As We May Think. *Atlantic Monthly*, 1945(July), 1945. This article also appeared in Life Magazine, 10 September 1945, pages 112–123.

[Bux92] William Buxton. Telepresence: Integrating Shared Task and Person Spaces. In *Proceedings of Graphics Interface ’92*, pages 123–129, 1992.

[Bux94] William Buxton. Two-handed Document Navigation. *XEROX Disclosure Journal*, 19(2):103–108, 1994.

[BW00] Gary Bishop and Greg Welch. Working in the Office of “Real Soon Now”. *IEEE Computer Graphics & Applications*, 20(4):76–78, 2000. Available at `<http://www.cs.unc.edu/~welch/oorsn.html>` (accessed 4 September 2003).

[Cal] GTCO Calcomp. `<http://www.gtco.com>` (accessed 13 December 2003). This company makes various tablets and digitizers for different applications.

[Car93a] Kathleen Carter. Computer Aided Design: Back to the Drawing Board. In *Proceedings of Creativity and Cognition, Loughborough, April 93*, 1993.

[Car93b] Kathleen Carter. Computer Aided Design: Back to the Drawing Board. Technical Report EPC-1993-107, Rank Xerox Research Centre, Cambridge, UK, 1993.

[Che02] Chaomei Chen. Editorial. *Information Visualization*, 1(1):1–4, 2002.

[CHWS88] Jack Callahan, Don Hopkins, Mark Weiser, and Ben Shneiderman. An Empirical Comparison of Pie vs. Linear Menus. In *Proceedings of CHI 88*, pages 95–100, 1988.

[CK01] Andy Cockburn and Bruce Kckenzie. 3D or not 3D? Evaluating the Effect of the Third Dimension in a Document Management System. In *Proceedings of CHI 2001*, pages 434–441, 2001.

[CM02] Andy Cockburn and Bruce McKenzie. Evaluating the Effectiveness of Spatial Memory in 2D and 3D Physical Virtual Environments. In *Proceedings of CHI 2002*, pages 203–210, 2002.

[CNSD93] Carolina Cruz-Neira, Daniel J. Sandin, and Thomas A. DeFanti. Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE. In *Proceedings of Siggraph 93*, pages 135–142, 1993.

[CPN] CPN Tools. <http://wiki.daimi.au.dk:8000/cpntools/_home.wiki> (accessed 4 September 2003).

[CRSS03] Tat-Jen Cham, James M. Rehg, Rahul Sukthankar, and Gita Sukthankar. Shadow Elimination and Occluder Light Suppression for Multi-Projector Displays. In *Proceedings of IEEE CVPR 2003*, 2003.

[CSR$^+$03] Mary Czerwinski, Greg Smith, Tim Regan, Brian Meyers, George Robertson, and gary Starkweather. Toward Characterizing the Productivity Benefits of Very Large Displays. In *Proceedings of IFIP Interact 2003* , pages 9–16, 2003.

[CSRS01] Tat-Jen Cham, Rahul Sukthankar, James M. Rehg, and Gita Sukthankar. Shadow Elimination and Occluder Light Suppression for Multi-Projector Displays. Technical Report CRLTR 2002/03, Compaq, 2001.

[CSWC01] Han Chen, Rahul Sukthankar, Grant Wallace, and Tat-Jen Cham. Calibrating Scalable Multi-Projector Displays Using Camera Homography Trees. In *Proceedings of IEEE CVPR 2001*, 2001.

[CSWL02] Han Chen, Rahul Sukthankar, Grant Wallace, and Kai Li. Scalable Alignment of Large-Format Multi-Projector Displays Using Camera Homography Trees . In *Proceedings of IEEE Visualization 2002*, pages 339–346, 2002.

[CVFM02] Duncan Cavens, Florian Vogt, Sidney Fels, and Michael Meitner. Interacting with the Big Screen: Pointers to Ponder. In *Extended Abstracts of CHI 2002*, pages 678–679, 2002.

[DC00] James Davis and Xing Chen. LumiPoint: Multi-User Laser-Based Interaction on Large Tiled Displays. Technical Report TR-2000-04, Stanford Computer Science Department, 2000.

[DC02] James Davis and Xing Chen. LumiPoint: Multi-User Laser-Based Interaction on Large Tiled Displays. *Displays*, 23(5):205–211, 2002.

[DHT00] Christian Heide Damm, Klaus Marius Hansen, and Michael Thomsen. Tool Support for Cooperative Object-Oriented Design: Gesture Based Modelling on an Electronic Whiteboard. In *Proceedings of CHI 2000*, pages 518–525, 2000.

[DL01] Paul Dietz and Darren Leigh. DiamondTouch: A Multi-User Touch Technology. In *Proceedings of UIST 2001*, pages 219–226, 2001.

[DW] NCSA Display-Wall-in-a-Box project. <http://www.ncsa.uiuc.edu/TechFocus/Deployment/DBox/> (accessed 4 September 2003).

[DX] Microsoft DirectX. <http://www.microsoft.com/windows/directx> (accessed 4 September 2003).

[EB] eBeam. <http://www.e-beam.com> (accessed 13 December 2003). This company makes an ultrasonic pen-tracking system.

[EBG$^+$92] Scott Elrod, Richard Bruce, Rich Gold, David Goldberg, Frank Halasz, William Janssen, David Lee, Kim McCall, Elin Pedersen, Ken Pier, John Tang, and Brent Welch. Liveboard: A Large Interactive Display Supporting Group Meetings, Presentations and Remote Collaboration. In *Proceedings of CHI 92*, pages 599–607, 1992.

[Egi88] Carmen Egido. Video Conferencing as a Technology to Support Group Work: a Review of its Failures . In *Proceedings of CSCW 88*, pages 13–24, 1988.

[EGR91] Clarence A. Ellis, Simon J. Gibbs, and Gail L. Rein. Groupware: Some Issues and Experiences. *Communications of the ACM*, 34(1), 1991.

[EH00] Ame Elliott and Marti A. Hearst. How Large Should a Digital Desk Be? Qualitative Results of a Comparative Study. In *Extended Abstracts of CHI 2000*, pages 165–166, 2000.

[EH02] Ame Elliott and Marti A. Hearst. A Comparison of the Affordances of a Digital Desk and Tablet for Architectural Tasks. *International Journal of Human-Computer Studies*, 56(2):173–197, 2002.

[EKLL03] Katherine M. Everitt, Scott R. Klemmer, Robert Lee, and James A. Landay. Two Worlds Apart: Bridging the Gap Between Physical and Virtual Media for Distributed Design Collaboration . In *Proceedings of CHI 2003*, pages 553–560, 2003. Also available as University of California, Berkeley, Technical Report UCB//CSD-02-1201.

[FB97] George Fitzmaurice and William Buxton. An Empirical Evaluation of Graspable User Interfaces: Towards Specialized, Space-Multiplexed Input. In *Proceedings of CHI 97*, pages 43–50, 1997.

[Fer92] Eugine S. Ferguson. *Engineering and the Mind's Eye*. MIT Press, 1992.

[FHHD90] Andrew Fountain, Wendy Hall, Ian Heath, and Hugh Davis. MICROCOSM: An Open Model for Hypermedia With Dynamic Linking. In *A. Rizk, N. Streitz and J. Andre (editors) Hypertext: Concepts, Systems and Applications, The Proceedings of The European Conference on Hypertext, INRIA, France*, 1990.

[FKS00] Susan R. Fussell, Robert E. Kraut, and Jane Siegel. Coordination of Communication: Effects of Shared Visual Context on Collaborative Work. In *Proceedings of CSCW 2000*, pages 21–30, 2000.

[FL00] Thomas Funkhouser and Kai Li. Large-Format Displays (introduction to special edition on large displays). In *IEEE Computer Graphics & Applications*, pages 20–21, 2000.

[Fre93] Steve M. G. Freeman. *An Architecture for Distributed User Interfaces*. PhD thesis, University of Cambridge Computer Laboratory, 1993.

[FS91] Steven Feiner and Ari Shamash. Hybrid User Interfaces: Breeding Virtually Bigger Interfaces for Physically Smaller Computers. In *Proceedings of UIST 91*, pages 9–17, 1991.

[GBL$^+$02] Jim Gemmel, Gordon Bell, Roger Lueder, Steven Drucker, and Curtis Wong. MyLifeBits: Fulfilling the Memex Vision. In *Proceedings of ACM Multimedia 2002*, pages 235–238, 2002.

[GHR95] Saul Greenberg, Stephen Hayne, and Roy Rada. *Groupware for Real-Time Drawing: A Designer's Guide*. McGraw-Hill, London, 1995. ISBN 0077078993.

[Gos86] A. Ardeshir Goshtasby. Piecewise Linear Mapping Functions for Image Registration. *Pattern Recognition*, 19(4):459–466, 1986.

[Gos88] A. Ardeshir Goshtasby. Image Registration by Local Approximation Methods. *Image and Vision Computing*, 6(4):255–261, 1988.

[Gos99] A. Ardeshir Goshtasby. Transformation Functions, 1999. This was a Siggraph 99 course on '2D and 3D Image Registration and Image Warping' and is available at `<http://www.siggraph.org/s99/conference/courses/>` (accessed 4 September 2003).

[GP02] Carl Gutwin and Raegan Penner. Improving Interpretation of Remote Gestures with Telepointer Traces. In *Proceedings of CSCW 2002*, pages 49–57, 2002.

[Gre96] Saul Greenberg. A Fisheye Text Editor for relaxed WYSIWIS Groupware. In *Conference Companion for CHI 96*, pages 212–213, 1996.

[Gru01] Jonathan Grudin. Partitioning Digital Worlds: Focal and Peripheral Awareness in Multiple Monitor Use. In *Proceedings of CHI 2001*, pages 458–465, 2001.

[GRWB92a] Saul Greenberg, Mark Roseman, David Webster, and Ralph Bohnet. Human and Technical Factors of Distributed Group Drawing Tools. *Interacting with Computers*, 4(1):364–392, 1992.

[GRWB92b] Saul Greenberg, Mark Roseman, David Webster, and Ralph Bohnet. Issues and Experiences Designing and Implementing Two Group Drawing Tools. In *Proceedings of Hawaii International Conference on System Sciences*, pages 138–150, 1992.

[GS] Ghostscript, an interpreter for the PostScript language. `<http://www.cs.wisc.edu/~ghost>` (accessed 4 September 2003).

[GSW01] François Guimbretière, Maureen Stone, and Terry Winograd. Fluid Interaction with High-Resolution Wall-size Displays. In *Proceedings of UIST 2001*, pages 21–30, 2001.

[Gui87] Yves Guiard. Asymmetric Division of Labor in Human Skilled Bimanual Action: The Kinematic Chain as a Model. *Journal of Motor Behaviour*, 19(4):486–517, 1987.

[Gui02] François Guimbretière. *Fluid Interaction with High-Resolution Wall-Sized Displays*. PhD thesis, Stanford University Department of Computer Science, 2002.

[Gui03] François Guimbretière. Paper Augmented Digital Documents. In *Proceedings of UIST 2003*, 2003.

[Gut99] Carl Gutwin. Traces: Visualizations of Interaction. Technical Report 99-1, Computer Science Department, University of Saskatchewan, 1999.

[Gut02] Carl Gutwin. Traces: Visualizing the Immediate Past to Support Group Interaction. In *Proceedings of Graphics Interface 2002*, pages 43–50, 2002.

[GW00] François Guimbretière and Terry Winograd. FlowMenu: Combining Command, Text, and Data Entry. In *Proceedings of UIST 2000*, pages 213–216, 2000.

[Har00] Beverly L. Harrison. E-Books and the Future of Reading. *IEEE Computer Graphics & Applications*, 2000(May/June):32–39, 2000.

[HEB$^+$01] Greg Humphreys, Matthew Eldridge, Ian Buck, Gordon Stoll, Matthew Everett, and Pat Hanrahan. WireGL: A Scalable Graphics System for Clusters. In *Proceedings of Siggraph 2001*, 2001.

[HEB$^+$02] Greg Humphreys, Matthew Eldridge, Ian Buck, Gordon Stoll, Matthew Everett, and Pat Hanrahan. Chromium: A Stream Processing Framework for Interactive Rendering on Clusters. In *Proceedings of Siggraph 2002*, pages 693–702, 2002.

[Hec89] Paul S. Heckbert. Fundamentals of Texture Mapping and Image Warping. Master's thesis, University of California, Berkeley, 1989. See pages 17–21 for projective mappings between homogeneous co-ordinate spaces.

[HH99] Greg Humphreys and Pat Hanrahan. A Distributed Graphics System for Large Tiled Displays. In *Proceedings of IEEE Visualization '99*, pages 215–224, 1999.

[HJS00] Mark Hereld, Ivan R. Judson, and Rick L. Stevens. Introduction to Building Projection-based Tiled Display Systems. *IEEE Computer Graphics & Applications*, 20(4):22–28, 2000.

[HL93] Josef Hoschek and Dieter Lasser. *Fundamentals of Computer Aided Geometric Design*. A K Peters Ltd., 1993. ISBN 1-56881-007-5.

[HPG93] Stephen Hayne, Mark Pendergast, and Saul Greenberg. *Saul Greenberg, Stephen Hayne and Roy Rada (editors), Groupware for Real-Time Drawing:A Designer's Guide*, chapter on Gesturing Through Cursors: Implementing Multiple Pointers in Group Support Systems, pages 63–80. McGraw-Hill, London, 1993. ISBN 0077078993.

[HPPK98] Ken Hinckley, Randy Pausch, Dennis Proffitt, and Neal F. Kassell. Two-Handed Virtual Manipulation. *ACM Transactions on Computer-Human Interaction*, 5(3):260–302, 1998.

[HS92] Jim Hollan and Scott Stornetta. Beyond Being There. In *Proceedings of CHI 92*, pages 119–125, 1992.

[HS00] Aldo Hoeben and Pieter Jan Stappers. Flicking through Page-based Documents with Thumbnail Sliders and Electronic Dog-ears. In *Extended Abstracts of CHI 2000*, pages 191–192, 2000.

[HZ00] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[IKG92] Hiroshi Ishii, Minoru Kobayashi, and Jonathan Grudin. Integration of Interpersonal Space and Shared Workspace: ClearBoard Design and Experiments. In *Proceedings of CSCW 92*, pages 33–42, 1992.

[IT] iText: A Free Java-PDF Library. <http://www.lowagie.com/iText> (accessed 4 September 2003).

[IU97] Hiroshi Ishii and Brygg Ullmer. Tangible Bits: Towards Seemless Interfaces Between People Bits and Atoms. In *Proceedings of CHI 97*, pages 234–241, 1997.

[JC86] D. Austin Henderson Jr. and Stuart K. Card. Rooms: The Use of Multiple Virtual Workspaces to Reduce Space Contention in a Window-Based Graphical User Interface. *ACM Transactions on Graphics*, 5(3):211–243, 1986.

[JJK$^+$93] W. Johnson, H. Jellinek, L. Klotz, R. Rao, and S. Card. Bridging the Paper and Electronic Worlds: The Paper User Interface . In *Proceedings of InterCHI 93*, pages 507–512, 1993.

[JRV$^+$89] Jeff Johnson, Teresa L. Roberts, William Verplank, David C. Smith, Charles Irby, Marian Beard, and Kevin Mackey. The Xerox Star: A Retrospective. *IEEE Computer*, 22(9):11–29, 1989. Reprinted in *Human Computer Interaction: Toward the year 2000*, R. M. Baeker et al., Morgan Kaufmann, pages 53–70, and available at <http://www.digibarn.com/friends/curbow/star/retrospect/> (accessed 4 September 2003).

[KB94] Gordon Kurtenbach and Willam Buxton. User Learning and Performance with Marking Menus . In *Proceedings of CHI 94*, pages 258–264, 1994.

[KBF$^+$95] Wolfgand Kruger, Christian Bohn, Bernd Frohlich, Heinrich Schth, Wolfgan Strauss, and Gerold Wesche. The Responsive Workbench: A Virtual Work Environment. *Computer*, 28(7):42–48, 1995.

[KE02] Scott Klemmer and Katherine Everitt. Bridging Physical and Electronic Media for Distributed Design Collaboration. In *Extended Abstracts of CHI 2002*, pages 878–879, 2002.

[Kea98] T. Alan Keahey. The Generalized Detail-In-Context Problem. In *Proceedings of IEEE Visualization 98*, 1998.

[KFBB97] Gordon Kurtenbach, George Fitzmaurice, Thomas Baudel, and William Buxton. The Design of a GUI Paradigm based on Tablets, Two-Hands and Transparency. In *Proceedings of CHI 97*, pages 35–42, 1997.

[KGF02] Robert K. Kraut, Darren Gergle, and Susan R. Fussel. The Use of Visual Information in Shared Visual Spaces: Informing the Development of Virtual Co-Presence. In *Proceedings of CSCW 2002*, pages 31–40, 2002.

[KK98] Motoki Kobayashi and Hideki Koike. EnhancedDesk: Integrating Paper Documents and Digital Documents. In *IEEE Proceedings of Asia Pacific Computer Human Interaction (APCHI '98)*, pages 57–62, 1998.

[KMB93] Paul Kabbash, I. Scott MacKenzie, and William Buxton. Human Performance Using Compuer Input Devices in the Preferred and Non-Preferred Hands. In *Proceedings of InterCHI 93*, pages 474–481, 1993.

[Kra94] Axel Kramer. Translucent Patches—Dissolving Windows. In *Proceedings of UIST 94*, pages 121–130, 1994.

[Kru83] Myron W. Krueger. *Artificial Reality*. Addison-Wesley, 1983.

[Kru93] Myron W. Krueger. Environmental Technology: Making the Real World Virtual. *Communications of the ACM*, 36(7):36–37, 1993.

[Kru02] Russell Kruger. Orientation and Gesture on Horizontal Displays. In *UbiComp 2002 Workshop on Collaboration with Interactive Walls and Tables*, 2002.

[KSK01] Hideki Koike, Yoichi Sato, and Yoshinori Kobayashi. Integrating Paper and Digital Information on EnhancedDesk: A Method for Real-Time Finger Tracking on Augmented Desk System. *ACM Transactions on Computer-Human Interaction*, 8(4):307–322, 2001.

[KXN$^+$02] Hideki Koike, Chen Xinlei, Yasuto Nakanishi, Kenji Oka, and Yoichi Sato. Two-Handed Drawing on Augmented Desk. In *Extended Abstracts of CHI 2002*, pages 760–761, 2002.

[LA94] Ying K. Leung and Mark D. Apperley. A Review and Taxonomy of Distortion-Oriented Presentation Techniques. *ACM Transactions on Computer-Human Interaction*, 1(2):126–160, 1994.

[LCC$^+$00] Kai Li, Han Chen, Yuqun Chen, Douglas W. Clark, Perry Cook, Stefanos Damianakis, Georg Essl, Adam Finkelstein, Thomas Funkhouser, Timothy Housel, Allison Klein, Zhiyan Liu, Emil Praun, Rudrajit Samanta, Ben Shedd, Jaswinder Pal Singh, George Tzanetakis, and Jiannan Zheng. Building and Using a Scalable Display Wall System. *IEEE Computer Graphics & Applications*, 20(4), 2000.

[LD02] Darren Leigh and Paul Dietz. DiamondTouch Characteristics and Capabilities. In *UbiComp 2002 Workshop on Collaboration with Interactive Walls and Tables*, 2002.

[LEF$^+$00] Mik Lamming, Marge Eldrige, Mike Flynn, Chris Jones, and David Pendlebury. Satchel: Providing Access to Any Document, Any Time, Anywhere. *ACM Transactions on Human-Computer Interaction*, 7(3):322–352, 2000.

[LHG92] Paul Luff, Christian Heath, and David Greatbatch. Tasks-in-Interaction: Paper and Screen Based Documentation in Collaborative Activity. In *Proceedings of CSCW 92*, pages 163–170, 1992.

[LJM98] Beth M. Lange, Mark A. Jones, and James L. Meyers. Insight Lab: An Immersive Team Environment Linking Paper, Displays and Data. In *Proceedings of CHI 98*, pages 550–557, 1998.

[LWLF01] Kok-Lim Low, Greg Welch, Anselmo Lastra, and Henry Fuchs. Life-Sized Projector-Based Dioramas. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST) 2001*, pages 93–101, 2001.

[LZB98] Andrea Leganchuk, Shumin Zhai, and William Buxton. Manual and Cognitive Benefits of Two-Handed Input: An Experimental Study. *Transactions on Human Computer Interaction*, 5(4):326–359, 1998.

[Mal83] Thomas W. Malone. How do People Organize Their Desks? Implications for the Design of Office Information Systems. *ACM Transactions on Office Information Systems*, 1(1):99–112, 1983. Now called ACM Transactions on Information Systems.

[Man93] Niall Mansfield. *The joy of X : an overview of the X Window system*. Addison-Wesley, 1993. ISBN 201565129.

[MBN$^+$02] Brad A. Myers, Rishi Bhatnagar, Jeffrey Nichols, Choon Hong Peck, Dave Kong, Robert Miller, and A. Chris Long. Interacting at a Distance: Measuring the Performance of Laser Pointers and Other Devices. In *Proceedings of CHI 2002*, pages 33–40, 2002.

[McM00] Thomas P. Moran and William can Melle. Tivoli: Integrating Structured Domain Objects into a Freeform Whiteboard Environment. In *Extended Abstracts of CHI 2000*, pages 20–21, 2000.

[MCvM97] Thomas P. Moran, Patrick Chiu, and William van Melle. Pen-based Interaction Techniques for Organising Material on an Electronic Whiteboard. In *Proceedings of UIST 97*, pages 45–54, 1997.

[MES$^+$01] Richard Mander, Daniel E.Rose, Gitta Salomon, Yin Yin Wong, Timothy Oren, Susan Booker, and Stephanie Houde. Method and Apparatus for Organizing Information in Computer System , 2001. US Patent 6,243,724, 5 June 2001. This is Apple Computer's patent for the Pile Metaphor.

[MG00] Preben Mogensen and Kaj Gronbek. Hypermedia in the Virtual Project Room. In *Proceedings of Hypertext 2000*, pages 113–122, 2000.

[MIEL99] Elizabeth D. Mynatt, Takeo Igarashi, W. Keith Edwards, and Anthony LaMarca. Flatland: New Dimensions in Office Whiteboards. In *Proceedings of CHI 99*, pages 346–353, 1999.

[MIEL00] Elizabeth D. Mynatt, Takeo Igarashi, W. Keith Edwards, and Anthony LaMarca. Designing an Augmented Writing Surface. *IEEE Computer Graphics & Applications*, 20(4):55–61, 2000.

[Mil68] George A. Miller. Psychology and Information. *American Documentation*, 19(3):286–289, 1968.

[Mim] Mimio from Virtual Ink. `<http://www.mimio.com>` (accessed 4 September 2003). An ultrasonic pen input device that can be attached to whiteboards.

[MMV$^+$01] Blair MacIntyre, Elizabeth D. Mynatt, Stephen Voida, Klaus M. Hansen, Joe Tullio, and Gregory M. Corso. Support for Multitasking and Background Awareness Using Interactive Peripheral Displays. In *Proceedings of UIST 2001*, pages 41–50, 2001.

[MRC91] Jock D. Mackinlay, George G. Robertson, and Stuart K. Card. Perspective Wall: Detail and Context Smoothly Integrated. In *Proceedings of CHI 91*, pages 173–179, 1991.

[MS03] Aditi Majumder and Rick Stevens. Color Non-Uniformity in Projection Based Displays: Analysis and Solutions, 2003. To appear in IEEE Transactions on Visualization and Computer Graphics.

[MSvM$^+$99] T. P. Moran, E. Saund, W. van Melle, et al. Design and Technology for Collaborage: Collaborative Collages of Information on Physical Walls. In *Proceedings of UIST 99*, pages 197–206, 1999.

[MSW92] Richard Mander, Gitta Salomon, and Yin Yin Wong. A 'Pile' Metaphor for Supporting Casual Organization of Information. In *Proceedings of CHI 92*, pages 627–634, 1992.

[MT] IS-900 MiniTrax Hand Tracker from Intersense. `<http://www.isense.com>` (accessed 4 September 2003). Intersense's IS-900 system uses ultrasound and inertial sensors in head and hand tracking devices.

[MT02] Carsten Magerkurth and Peter Tandler. Interactive Walls and Handheld Devices – Applications for a Smart Environment. In *UbiComp 2002 Workshop on Collaboration with Interactive Walls and Tables*, 2002.

[MvMC98] Thomas P. Moran, William van Melle, and Patrick Chio. Tailorable Domain Objects as Meeting Tools for an Electronic Whiteboard. In *Proceedings of CSCW 98*, pages 295–304, 1998.

[My] Myrinet. `<http://www.myri.com/>` (accessed 4 September 2003). Myrinet is a high-performance, packet-communication and switching technology used to interconnect clusters of PCs.

[Nag84] John Nagle. Internet RFC 896, 1984. Available at `<http://www.ietf.org>` (accessed 4 September 2003).

[NM] Microsoft NetMeeting. `<http://www.microsoft.com/windows/netmeeting/>` (accessed 4 September 2003).

[NO92] Adrian Nye and Tim O'Reilly. *X Toolkit Intrinsics Programming Manual: OSF/Motif 1.1 edition for X11, release 5*. O'Reilly & Associates, 1992. ISBN 1565920139.

[Nor88] Donald A. Norman. *The Psychology of Everyday Things*. Basic Books, 1988. ISBN 0465067093.

[Nor93] Donald A. Norman. *Things That Make Us Smart: Defending Human Attributes in the Age of the Machine*. Addison-Wesley, New York, 1993.

[NOT$^+$96] Masaki Nakagawa, T. Oguni, T.Yoshino, K. Horiba, and S. Sawada. Interactive Dynamic Whiteboard for Educational Applications. In *Proceedings of Virtual Systems and Multimedia (VSMM) 96*, pages 479–484, 1996.

[Num] Numonics Corporation. `<http://www.numonics.com>` (accessed 4 September 2003). This company makes digitizers for CAD and GIS applications.

[NW92] William Newman and Pierre Wellner. A Desk Supporting Computer-based Interaction with Paper Documents. In *Proceedings of CHI 92*, pages 587–562, 1992.

[Nye90] Adrian Nye. *X Protocol Reference Manual for Version 11 of the X Window System*. O'Reilly, 1990.

[O'R94] Joseph O'Rourke. *Computational Geometry in C*. Cambridge University Press, 1994.

[Orl03] Andrew Orlowski. Deep Inside Apple's Piles. *The Register*, 2003(23 April), 2003. Available at `<http://www.theregister.co.uk/content/39/30360.html>` (accessed 4 September 2003).

[OS97] Kenton O'Hara and Abigail Sellen. A Comparison of Reading Paper and On-line Documents. In *Proceedings of CHI 97*, pages 335–342, 1997.

[OSK02] Kenji Oka, Yoichi Sato, and Hideki Koike. Real-time Tracking of Multiple Fingertips and Gesture Recognition for Augmented Desk Interface Systems. In *Proceedings of Fifth IEEE International Conference on Automatic Face and Gesture Recognition (FG) 2002*, pages 429–434, 2002.

[Par02] Joseph A. Paradiso. Several Sensor Approaches that Retrofit Large Surfaces for Interactivity. In *UbiComp 2002 Workshop on Collaboration with Interactive Walls and Tables*, 2002.

[PDF] Adobe Portable Document Format (PDF). `<http://www.adobe.com/products/acrobat/adobepdf.html>` (accessed 13 December 2003).

[Peg] Pegasus Technologies. `<http://www.pegatech.com>` (accessed 4 September 2003). This company make the Natural Writing Board, an ultrasonic pen tracking system for whiteboards.

[Pem95] Steven Pemberton. Metaphorically Speaking. *SIGCHI Bulletin*, 27(4):96, 1995.

[Pic] The Piccolo Project. `<http://www.cs.umd.edu/hcil/piccolo/>` (accessed 4 September 2003).

[PIHP01] James Patten, Hiroshi Ishii, Jim Hines, and Gian Pangaro. Sensetable: A Wireless Object Tracking Platform for Tangible User Interfaces. In *Proceedings of CHI 2001*, pages 253–260, 2001.

[Pin01a] Claudio Pinhanez. Augmenting Reality with Projected Interactive Displays. In *Proceedings of International Symposium on Virtual and Augmented Architecture (VAA 2001)*, 2001.

[Pin01b] Claudio Pinhanez. The Everywhere Displays Projector: A Device to Create Ubiquitous Graphical Interfaces. In *Proceedings of Ubiquitous Computing (UbiComp) 2001*, pages 315–331, 2001.

[PKL$^+$01] Claudio Pinhanez, Rick Kjeldsen, Anthony Levas, Gopal Pingali, Jacob Hartman, Vivek Kwatra, Mark Podlaseck, and Paul Chou. Transforming Surfaces into Touch-Screens. Technical Report RC22273 (W0112-016), IBM T J Watson Reserch Center, 2001.

[PKL$^+$02] Claudio Pinhanez, Frederik Kjeldsen, Anthony Levas, Gopal Pingali, Mark Podlaseck, and Paul Chou. Ubiquitous Interactive Graphics. Technical Report RC22495 (W0205-143), IBM T J Watson Reserch Center, 2002.

[PMMH93] Elin Pedersen, Kim McCall, Thomas P. Moran, and Frank G. Halasz. Tivoli: An Electronic Whiteboard for Informal Workgroup Meetings. In *Proceedings of InterCHI 93*, pages 391–398, 1993.

[Pol] Polhemus. <http://www.polhemus.com> (accessed 4 September 2003). This company makes the Fastrak tethered magnetic tracking system.

[PPA$^+$03] Mark Podlaseck, Claudio Pinhanez, Nancy Alvarado, Margaret Chan, and Elisa Dejesus. On Interfaces Projected onto Real-World Objects. In *CHI 2003 Short Papers*, 2003.

[PPL$^+$02] Gopal Pingali, Claudio Pinhanez, Tony Levas, Rick Kjeldsen, and Mark Podlaseck. User-Following Displays. In *Proceedings of IEEE International Conference on Multimedia and Expo (ICME) 2002*, 2002.

[PPL$^+$03] Gopal Pingali, Claudio Pinhanez, Anthony Levas, Rick Kjeldsen, Mark Podlaseck, Han Chen, and Noi Sukaviriya. Steerable Interfaces for Pervasive Computing Spaces. In *Proceedings of the First IEEE Conference on Pervasive Computing and Communications (PerCom 2003)*, pages 315–322, 2003.

[PRI02] James Patten, Ben Recht, and Hiroshi Ishii. Audiopad: A Tag-Based Interface for Musical Performance. In *Proceedings of New Interfaces for Musical Expression (NIME) 2002*, 2002.

[PT02] Thorsten Prante Peter Tandler, Norbert A. Streitz. Roomware—Moving Toward Ubiquitous Computers. *IEEE Micro*, 2002(Nov):36–47, 2002. Available at <http://www.ipsi.fhg.de/ambiente/> (accessed 4 September 2003).

[PTVF92] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 1992. Available from <http://www.nr.com/> (accessed 4 September 2003).

[PTVF02] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C++*. Cambridge University Press, 2002.

[PW] PowerWall (University of Minnesota). <http://www.lcse.umn.edu/research/powerwall/powerwall.html> (accessed 4 September 2003).

[Ras00] Jef Raskin. *The Humane Interface*. Addison Wesley, 2000.

[RBW01] Holger Regenbrecht, Gregory Baratoff, and Michael Wagner. A tangible AR desktop environment. *Computers & Graphics*, 25(5):755–763, 2001.

[RBY$^+$99] Ramesh Raskar, Michael Brown, Ruigang Yang, Wei-Chao Chen, Greg Welch, Herman Towles, Brent Seales, and Henry Fuchs. Multi-Projector Displays Using Camera-Based Registration. In *Proceedings of IEEE Visualization '99*, pages 161–168, 1999.

[RCL98] George Robertson, Mary Czerwinski, and Kevin Larson. Data Mountain: Using Spatial Memory for Document Management. In *Proceedings of UIST 98*, pages 153–162, 1998.

[RCM89] George G. Robertson, Stuart K. Card, and Jock D. Mackinlay. The Cognitive Coprocessor Architecture. In *Proceedings of UIST 89*, pages 10–18, 1989.

[RCMS03] Tim Regan, Mary Czerwinski, Brian Meyers, and Greg Smith. Bumping Windows Between Monitors. Technical Report MSR-TR-2003-13, Microsoft Research, 2003.

[RDS02] Daniel M. Russell, Celmens Drews, and Alison Sue. Social Aspects of Using Large Public Interactive Displays for Collaboration. In *Proceedings of UbiComp 2002*, pages 229–236, 2002.

[Rek97] Jun Rekimoto. Pick-and-Drop: A Direct Manipulation Technique For Multiple Computer Environments. In *Proceedings of UIST 97*, pages 31–39, 1997.

[Rek98] Jun Rekimoto. A Multiple Device Approach for Supporting Whiteboard-Based Interactions. In *Proceedings of CHI 98*, pages 344–351, 1998.

[Rek00] Jun Rekimoto. Multiple-Computer User Interfaces: “Beyond the Desktop” Direct Manipulation Environments. In *Extended Abstracts of CHI 2000*, pages 6–7, 2000.

[Rek02] Jun Rekimoto. SmartSkin: An Infrastructure for Freehand Manipulation on Interactive Surfaces. In *Proceedings of CHI 2002*, pages 113–120, 2002.

[RFC$^+$02] James M. Rehg, Matthew Flagg, Tat-Jen Cham, Rahul Sukthankar, and Gita Sukthankar. Projected Light Displays Using Visual Feedback. In *Proceedings of International Conference on Control, Automation, Robotics, and Vision (ICARCV) 2002*, 2002.

[RG92] Mark Roseman and Saul Greenberg. GroupKit: A Groupware Toolkit for Building Real-Time Conferencing Applications. In *Proceedings of CSCW 92*, pages 43–50, 1992.

[RM93a] George G. Robertson and Jock D. Mackinlay. The Document Lens. In *Proceedings of UIST 93*, pages 101–108, 1993.

[RM$^+$93b] Daniel E. Rose, Richard Mander, , Tim Oren, Dulce B. Ponceleón, Gitta Salomon, and Yin Yin Wong. Content Awareness in a File System Interface: Implementing the ‘Pile’ Metaphor for Organizing Information. In *Proceedings of SIGIR 93*, pages 260–269, 1993.

[Rob95] Peter Robinson. Virtual offices, 1995. Proceedings of Royal Society discussion meeting on Virtual Reality, July 1995, British Telecom Publication number SRD/R5/1.

[Rob99] Peter Robinson. Digital Manuscripts and Electronic Publishing. *Editio*, 1999(Autumn):337–346, 1999. ISBN 3-484-29513-9.

[RR01] John A. Robinson and Charles Robertson. The LivePaper System: Augmenting Paper on an Enhanced Tabletop. *Computers & Graphics*, 25(5):731–743, 2001.

[RRHT03] Tom Rodden, Yvonne Rogers, John Halloran, and Ian Taylor. Designing Novel Interactional Workspaces to Support Face to Face Consultations. In *Proceedings of CHI 2003*, pages 57–64, 2003.

[RS99] Jun Rekimoto and Masanori Saitoh. Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments. In *Proceedings of CHI 99*, pages 378–385, 1999.

[RSFWH98] Tristan Richardson, Quentin Stafford-Fraser, Kenneth R. Wood, and Andy Hopper. Virtual Network Computing. *IEEE Internet Computing*, 2(1):33–38, 1998.

[RSW$^+$97] Peter Robinson, Dan Sheppard, Richard Watts, Robert Harding, and Steve Lay. A Framework for Interacting with Paper. In *Proceedings of Eurographics ’97 (Computer Graphics Forum)*, 1997.

[RvBB$^+$03] Ramesh Raskar, Jeroen van Baar, Paul Beardsley, Thomas Willwacher, Srinivas Rao, and Clifton Forlines. iLamps: Geometrically Aware and Self-Configuring Projectors. In *Proceedings of Siggraph 2003*, 2003.

[RvDR$^+$00] George Robertson, Maarten van Dantzich, Daniel Robbins, Mary Czerwinski, Ken Hinckley, Kirsten Risden, David Thiel, and Vadim Gorokhovsky. The Task Gallery: A 3D Window Manager. In *Proceedings of CHI 2000*, pages 494–501, 2000.

[RWC$^+$98] Ramesh Raskar, G. Welch, M. Cutts, M. Lake, L. Stesin, and H. Fuchs. The Office of the Future: A Unified Approach to Image-Based Modelling and Spatially Immersive Displays. In *Proceedings of Siggraph 98*, pages 179–188, 1998.

[RWF98] Ramesh Raskar, G. Welch, and H. Fuchs. Seamless Projection Overlaps using Image Warping and Intensity Blending. In *Proceedings of Virtual Systems and Multimedia (VSMM) 98*, 1998.

[RZW02] Ramesh Raskar, Remo Ziegler, and Thomas Willwacher. Cartoon Dioramas in Motion. In *Proceedings of the Second International Symposium on Non-Photorealistic Animation and Rendering (NPAR 2002)*, 2002. Also available as Mitsubishi Electric Research Laboratories (MERL) technical report TR-2002-28.

[SB94] Manojit Sarkar and Marc H. Brown. Graphical Fisheye Views. *Communications of the ACM*, 1994(Dec):73–83, 1994.

[SBD99] Jason Stewart, Benjamin B. Bederson, and Allison Druin. Single Display Groupware: A Model for Co-present Collaboration. In *Proceedings of CHI 99*, pages 286–293, 1999.

[SCS01] Rahul Sukthankar, Tat-Jen Cham, and Gita Sukthankar. Dynamic Shadow Elimination for Multi-Projector Displays. In *Proceedings of IEEE CVPR 2001*, 2001.

[ScT] Screen Technology. `<http://www.screentechnology.com/>` (accessed 4 September 2003).

[SF96] J. Quentin Stafford-Fraser. *Video Augmented Environments*. PhD thesis, University of Cambridge Computer Laboratory, 1996.

[SFR96] Quentin Stafford-Fraser and Peter Robinson. BrightBoard: A Video-Augmented Environment. In *Proceedings of CHI 96*, pages 134–141, 1996.

[SGH$^+$99] Norbert A. Streitz, Jrg Geiler, Torsten Holmer, Shinichi Konomi, Christian Mller-Tomfelde, Wolfgang Reischl, Petra Rexroth, Peter Seitz, and Ralf Steinmetz. i-LAND: A Interactive Landscape for Creativity and Innovation. In *Proceedings of CHI 99*, pages 120–127, 1999.

[SGHH94] Norbert A. Streitz, Jorg Geissler, Jorg M. Haake, and Jeroen Hol. DOLPHIN: Integrated Meeting Support across Local and Remote Desktop Environments and Liveboards. In *Proceedings of CSCW 94*, pages 345–358, 1994.

[SGP98] Bill N. Schilit, Gene Golovchinsky, and Morgan N. Price. Beyond Paper: Supporting Active Reading with Free Form Digital Ink Annotations. In *Proceedings of CHI 98*, pages 249–256, 1998.

[SH97] Abigail Sellen and Richard Harper. Paper as an Analytic Resource for the Design of New Technologies. In *Proceedings of CHI 97*, pages 319–326, 1997.

[SHC$^+$95] Stephen A. R. Scrivener, David Harris, Sean M. Clark, Todd Rockoff, and Michael Smyth. *Saul Greenberg, Stephen Hayne and Roy Rada (editors), Groupware for Real-Time Drawing:A Designer's Guide*, chapter on Designing at a Distance via Real-time Designer-to-designer Interaction, pages 5–23. McGraw-Hill, London, 1995. ISBN 0077078993.

[SIKV82] David C. Smith, Charles Irby, Ralph Kimball, and Bill Verplank. Designing the Star User Interface. *Byte Magazine*, 1982(April), 1982. Reprinted in *Human-Computer Interaction*, J. Preece and L. Keller, pages 237–259, Prentice Hall, 1990.

[Sim99] Eero Simoncelli. Least Squares Optimization, 1999. `<http://www.cns.nyu.edu/~eero/math-tools/leastSquares.pdf>` (accessed 4 September 2003).

[SKK00] Yoichi Sato, Yoshinori Kobayashi, and Hideki Koike. Fast Tracking of Hands and Fingertips in Infrared Images for Augmented Desk Interface. In *Proceedings of Fourth IEEE International Conference on Automatic Face and Gesture Recognition (FG 2000)*, pages 462–467, 2000.

[SLV03] Chia Shen, Neal Lesh, and Frederic Vernier. Personal Digital Historian: Story Sharing Around the Table. *ACM Interactions*, 2003(March):15–22, 2003.

[SPK$^+$03a] Noi Sukaviriya, Mark Podlaseck, Rick Kjeldsen, Anthony Levas, Gopal Pingali, and Claudio Pinhanez. Augmenting a Retail Environment Using Steerable Interactive Displays . In *Extended Abstracts of CHI 2003*, pages 978–979, 2003.

[SPK$^+$03b] Noi Sukaviriya, Mark Podlaseck, Rick Kjeldsen, Anthony Levas, Gopal Pingali, and Claudio Pinhanez. Embedding Interactions in a Retail Store Environment: The Design and Lessons Learned. In *Proceedings of Ninth IFIP International Conference on Human-Computer Interaction (INTERACT 2003)*, 2003.

[SPK$^+$03c] Noi Sukaviriya, Mark Podlaseck, Rick Kjeldsen, Anthony Levas, Gopal Pingali, and Claudio Pinhanez. Ubiquitous Interactive Displays in a Retail Environment. In *Siggraph 2003 sketches*, 2003.

[SPMT+02] Norbert Streitz, Thorsten Prante, Christian Mller-Tomfelde, Peter Tandler, and Carsten Magerkurth. Roomware—The Second Generation. In *Extended Abstracts of CHI 2002*, pages 506–507, 2002. Available at <http://www.ipsi.fhg.de/ambiente/> (accessed 4 September 2003).

[SS97] Kishore Swaminathan and Steve Sato. Interaction Design for Large Displays. *Interactions*, 4(1):15–24, 1997.

[SSM00] Rahul Sukthankar, Robert G. Stockton, and Matthew D. Mullin. Self-Calibrating Camera-Assisted Presentation Interface. In *Proceedings of ICARCV 2000*, 2000.

[SSM01] Rahul Sukthankar, Robert G. Stockton, and Matthew D. Mullin. Smarter Presentations: Exploiting Homography in Camera-Projector Systems. In *IEEE Proceedings of ICCV 2001*, pages 247–253, 2001.

[SSTR93] Manojit Sarkar, Scott S. Snibbe, Oren J. Tversky, and Steven P. Reiss. Stretching the Rubber Sheet: A Metaphor for Large Layouts on Small Screens. In *Proceedings of UIST 93*, pages 81–91, 1993.

[ST] SMART Technologies Inc. <http://www.smarttech.com> (accessed 13 December 2003). This company makes the SMART Board, a touch sensitive whiteboard.

[Sto] Maureen C. Stone. Color Balancing Experimental Projection Displays. Submitted to 9th IST/SID Color Imaging Conference. Available at <http://graphics.stanford.edu/papers/dlp/> (accessed 24 Sept 2003).

[Sto01] Maureen C. Stone. Color and Brightness Appearance Issues in Tiles Displays. *IEEE Computer Graphics & Applications*, 21(5):58–66, 2001.

[TB] TeamBoard. <http://www.teamboard.com> (accessed 4 September 2003). This Company makes a touch sensitive whiteboard.

[TC03] Desney Tan and Mary Czerwinski. Examining HCI Attributes of Peripheral and Physically Large Displays. In *CHI 2003 Workshop on Providing Elegant Peripheral Awareness*, 2003.

[TCY+02] Herman Towles, Wei-Chao Chen, Ruigang Yang, Sang-Uok Kum, Henry Fuchs, Nikhil Kelshikar, Jane Mulligan, Kostas Daniilidis, Loring Holden, Bob Seleznik, Amela Sadagic, and Jaron Lanier. 3D Tele-Collaboration Over Internet2. In *International Workshop on Immersive Telepresence (ITP 2002) at ACM Multimedia*, 2002.

[TGSP03] Desney S. Tan, Darren Gergle, Peter G. Scupelli, and Randy Pausch. With Similar Visual Angles, Large Displays Improve Spatial Performance. In *Proceedings of CHI 2003*, pages 217–224, 2003.

[THSK01] Jonathan Trevor, David M. Hilbert, Bill N. Schilit, and Tzu Khiau Koh. From Desktop to Phonetop: A UI for Web Interaction on Very Small Devices. In *Proceedings of UIST 2001*, pages 121–130, 2001.

[THY+94] Masayuki Tani, Masato Horita, Kimiya Yamaashi, Koichiro Tanikoshi, and Masayasu Futakawa. Courtyard: Integrating Shared Overview on a Large Screen and Per-User Detail on Individual Screens. In *Proceedings of CHI 94*, pages 44–50, 1994.

[TIS+03] Yasuhisa Tokuda, Shinsuke Iwasaki, Yoichi Sato, Yasuto Nakanishi, and Hideki Koike. Ubiquitous Display for Dynamically Changing Environments. In *Extended Abstracts of CHI 2003*, pages 976–977, 2003.

[TL88] John C. Tang and Larry J. Leifer. A Framework for Understanding the Workspace Activity of Design Teams. In *Proceedings of CSCW 88*, pages 244–249, 1988.

[TM91] John C. Tang and Scott L. Minneman. VideoDraw: A Video Interface for Collaborative Drawing. *ACM Transactions on Information Systems*, 9(2):170–184, 1991.

[TM03] Quan T. Tran and Elizabeth D. Mynatt. What Was I Cooking? Towards Deja Vu Displays of Everyday Memory, 2003. Unpublished paper. Available at <http://www.cc.gatech.edu/fce/ecl/projects/cooking/> (accessed 4 September 2003).

[TPSP02] Desney S. Tan, Randy Pausch, Jeanine K. Stefanucci, and Dennis R. Proffitt. Kinesthetic Cues Aid Spatial Memory. In *Extended Abstracts of CHI 2002*, pages 806–807, 2002.

[TSB$^+$01] Naoya Takao, Jianbo Shi, Simon Baker, Iain Matthews, and Bart Nabbe. Tele-Graffiti: A Pen and Paper-Based Remote Sketching System. In *8th International Conference on Computer Vision (ICCV) 2001*, page 750, 2001.

[TSB02] Naoya Takao, Jianbo Shi, and Simon Baker. Tele Graffiti. Technical Report CMU-RI-TR-02-10, Carnegie Mellon University, 2002.

[TW80] Gabor P. Torok and Andrew B. White. Remote Chalkboard Automatic Cursor, 1980. US Patent 4,317,956, 10 November 1980.

[UI97] Brygg Ullmer and Hiroshi Ishii. The metaDESK: Models and Prototypes for Tangible User Interfaces. In *Proceedings of UIST 97*, pages 223–232, 1997.

[UI99] John Underkoffler and Hiroshi Ishii. Urp: A Luminous-Tangible Workbench for Urban Planning and Design. In *Proceedings of CHI 99*, pages 386–393, 1999.

[VDMC02] Stephen Voida, Elizabeth D.Mynatt, Blair MacIntyre, and Gregory M. Corso. Integrating Virtual and Physical Context to Support Knowledge Workers. *IEEE Pervasive Computing*, 1(3):73–79, 2002.

[VdSPvG03] Jouke C. Verlinden, A. de Smit, A. W. J. Peeters, and M. H. van Gelderen. Development of a Flexible Augmented Prototyping System. In *Proceedings of the 11th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG 2003)*, pages 496–503, 2003.

[VNC] Virtual Network Computing (VNC), a remote display system. `<http://www.realvnc.com>` (accessed 13 December 2003).

[Wac] Wacom. `<http://www.wacom.com>` (accessed 13 December 2003).

[Web] Webster. `<http://www.websterboards.com/>` (accessed 13 December 2003).

[Wei91] Mark Weiser. The Computer for the 21st Century. *Scientific American*, 1991(Sept):94–104, 1991.

[Wel] Greg Welch. The Office of "Real Soon Now". `<http://www.cs.unc.edu/~welch/oorsn.html>` (accessed 4 September 2003).

[Wel93a] Pierre D. Wellner. Interacting with Paper on the DigitalDesk. *Communications of the ACM*, 36(7):87–97, 1993.

[Wel93b] Pierre D. Wellner. Self Calibration for the DigitalDesk. Technical Report EPC-1993-109, Rank Xerox Research Centre, Cambridge, UK, 1993.

[Wel94] Pierre D. Wellner. *Interacting with Paper on the DigitalDesk*. PhD thesis, University of Cambridge Computer Laboratory, 1994.

[WH01] Steve Whittaker and Julia Hirschberg. The Character, Value and Management of Personal Paper Archives. *ACM Transactions on Computer-Human Interaction*, 8(2):150–170, 2001.

[WND99] Mason Woo, Jackie Neider, and Tom Davis. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 1.2*. Addison Wesley, 1999. ISBN 0201604582.

[WR93] Catherine G. Wolf and James R. Rhyne. Gesturing with Shared Drawing Tools. In *Comference Comanion for InterCHI 93*, pages 137–138, 1993.

[WSK$^+$00] Bin Wei, Claudio Silva, Eleftherios Koutsofios, Shankar Krishnan, and Stephen North. Visualization Research with Large Displays. *IEEE Computer Graphics and Applications*, 2000(July/August):2–6, 2000.

[WSS97] Lynn D. Wilcox, Bill N. Schilit, and Nitin Sawhney. Dynomite: A Dynamically Organised Ink and Audio Notebook. In *Proceedings of CHI 97*, pages 186–193, 1997.

[WT] Wintab. <http://www.pointing.com> (accessed 4 September 2003). An industry standard software interface for pointing devices.

[WWHW96] Benjamin Watson, Neff Walker, Larry F. Hodges, and Aileen Worden. Effectiveness of Peripheral Level of Detail Degradation When Used with Head-Mounted Displays. Technical Report 96-04, GVU Center, Georgia Institute of Technology, 1996.

[YGH$^+$01] Ruigang Yang, David Gotz, Justin Hensley, Herman Towles, and Michael S. Brown. PixelFlex: A Reconfigurable Multi-Projector Display System. In *Proceedings of IEEE Visualization 2001*, pages 167—174, 2001.

[YKN$^+$02] Ruigang Yang, Celso Kurashima, Andrew Nashel, Herman Towles, Anselmo Lastra, and Henry Fuchs. Creating Adaptive Views for Group Video Teleconferencing – An Image-Based Approach. In *International Workshop on Immersive Telepresence (ITP 2002) at ACM Multimedia*, 2002.

[Yod97] Lars Yoder. The Digital Display Technology of the Future. In *Proceedings of InfoComm 97, 5–7 June, Los Angeles, California, USA*, 1997. Available at <http://www.dlp.com> (accessed 4 September 2003).

Appendices

A Centroid of a set of points

The centroid of a set of points achieves the minimum sum of squared distances from the points.

Proof: Take a set of points $\mathbf{p}_i$ where $i \in [1, n]$. Their centroid $\overline{\mathbf{p}}$ is given by

$$\overline{\mathbf{p}} \quad = \quad \frac{1}{n}\sum_{i=1}^{n}\mathbf{p}_i \ .$$

We can express the sum of square distances from any point $\mathbf{q}$ to the points $\mathbf{p}_i$ as

$$\sum_{i=1}^{n}|\mathbf{p}_i - \mathbf{q}|^2 \quad = \quad \sum_{i=1}^{n}|\mathbf{p}_i - \overline{\mathbf{p}}|^2 + n|\mathbf{q} - \overline{\mathbf{p}}|^2 \ , \tag{1}$$

since,

$$\begin{aligned}
\sum|\mathbf{p}_i - \overline{\mathbf{p}}|^2 + n|\mathbf{q} - \overline{\mathbf{p}}|^2 &= \sum|\mathbf{p}_i|^2 - 2\overline{\mathbf{p}}\cdot\sum\mathbf{p}_i + n|\overline{\mathbf{p}}|^2 + n|\mathbf{q}|^2 - 2n\overline{\mathbf{p}}.\mathbf{q} + n|\overline{\mathbf{p}}|^2 \\
&= \sum|\mathbf{p}_i|^2 - 2n|\overline{\mathbf{p}}|^2 + n|\overline{\mathbf{p}}|^2 + n|\mathbf{q}|^2 - 2\mathbf{q}\cdot\sum\mathbf{p}_i + n|\overline{\mathbf{p}}|^2 \\
&= \sum|\mathbf{p}_i|^2 - 2\mathbf{q}\cdot\sum\mathbf{p}_i + n|\overline{\mathbf{q}}|^2 \\
&= \sum|\mathbf{p}_i - \mathbf{q}|^2 \ .
\end{aligned}$$

From Equation (1) it is obvious that the sum is minimized when $\mathbf{q} = \overline{\mathbf{p}}$.

B Escritoire client hardware

I assembled two instances of the Escritoire hardware, one at the Computer Laboratory in Cambridge and one at Thales Research and Development in Reading. The system at Cambridge, on which I developed the software, achieves 30 frames per second when moving a tile with approximately 300,000 pixels across the desk. This is with a 1024×768 pixel foveal projector using a 1024×1024 texture and a 640×480 periphery projector using a 512×512 texture. Performance is dependent on the time to receive new information from the server, and to update the textures. The time for warping the textures is minimal. The hardware used in the two systems is listed below.

Computer Laboratory

Processor	AMD Athlon 900 MHz
Memory	256 MB
Fovea video card	Matrox Millennium G400 AGP
Periphery video card	Matrox Millennium G450 PCI
Fovea projector	Proxima DX3, 1024×768 native resolution
Periphery projector	Proxima 9250+, 800×600 native resolution
Network to server	100 Mb/s Ethernet
Ultrasonic pen	Mimio, by Virtual Ink
Digitizer	Summagrid V, 36×48 inch active area, standard accuracy (±0.25mm), cordless stylus
6DOF tracker	Polhemus Fastrak magnetic tracker
Operating system	Microsoft Windows 2000
Graphics API	Microsoft DirectX 9.0

Thales

Processor	Intel Pentium 4, 2.4 GHz
Memory	256 MB
Video card	Matrox Millennium G450 DualHead
Projectors	Sanyo PLC-XW20A, 1024×768 native resolution, 1100 ANSI lumens
Network to server	100 Mb/s Ethernet
Ultrasonic pen	Mimio, by Virtual Ink
Digitizer	GTCO Calcomp DrawingBoard IV, 36×48 inch active area, standard accuracy (±0.25mm), cordless stylus
6DOF tracker	no
Operating system	Microsoft Windows XP Professional
Graphics API	Microsoft DirectX 8.1

C Using Ghostscript on PDF Documents

Before a PDF document is displayed on the Escritoire it must be converted from its native vector format into bitmap form so that it can be transmitted to the client program, written into a texture on the video card, and warped before being projected onto the desk. To do this conversion the server calls Ghostscript, an open source PDF and Postscript processor available from the University of Wisconsin `<http://www.cs.wisc.edu/~ghost/>`. I have used the version for Windows with the following command line:

```
gswin32c.exe
  -dQUIET
  -dBATCH
  -dNOPAUSE
  -dTextAlphaBits=2
  -dGrapicsAlphaBits=2
  -sDEVICE=bmp16m
  -sPAPERSIZE=a4
  -r <resolution>
  -sOutputFile=%03d.bmp
  <pdf file>
```

The first three options simply make the program do its processing without producing any output for the user, or any error messages if there is a problem. The next two select the maximum level of alpha blending, which especially improves the appearance of text rendered at a fairly low resolution. The `sDEVICE` option selects 16 bits-per-pixel bitmap output—32 bits-per-pixel output would double the size of the data to be stored and transmitted for a negligible visual improvement. The `sPAPERSIZE` option selects A4 paper. The `r` option sets the resolution of the rendering in dots per inch, which is set to a sensible value at the server, then the textures on the client are drawn at the appropriate size so that the pages on the desk appear life-sized. Fixing the resolution at the server rather than having a different resolution at each client makes it unnecessary to scale the bitmap data before it is copied into the foveal texture at the client, which reduces artifacts. The `sOutputFile` option indicates that that pages of the PDF document should be placed in bitmap files with successive numeric names, and the final option indicates the PDF file to process.

Index