A COLLABORATIVE WEARABLE SYSTEM

WITH REMOTE POINTING

by

MARTIN BAUER

A THESIS

Presented to the Department of Computer
and Information Science
and the Graduate School of the University of Oregon
in partial fulfillment of the requirements
for the degree of
Master of Science

December 1998

"A Collaborative Wearable System with Remote Pointing", a thesis prepared by Martin Bauer in partial fulfillment of the requirements for the Master of Science degree in the Department of Computer and Information Science.

This thesis has been approved and accepted by:

_____

Dr. Zary Segall

_____

Date

Committee in charge: Dr. Zary Segall

Accepted by:

_____

Dean of the Graduate School

An Abstract of the Thesis of

Martin Bauer for the degree of Master of Science

in the Department of Computer and Information Science

to be taken December 1998

Title: A COLLABORATIVE WEARABLE SYSTEM WITH REMOTE POINTING

Approved: _____

Dr. Zary Segall

This thesis presents the prototype of a collaborative wearable computer system whose main focus is on remote pointing. Remote pointing enables a remote user to point at objects in the physical environment of another user who is equipped with a wearable computer.

The central question for our research is how effective and efficient remote pointing is for collaboration and communication in a wearable context.

After discussing the design and implementation of the system, we present the results of a usability study evaluating the utility of remote pointing. We pay special attention to its influence on the communication between two workers collaborating on a simple task.

We show that users readily accept remote pointing as a tool and utilize it extensively. Since it provides a common point of focus, their communication becomes more efficient.

CURRICULUM VITA

NAME OF AUTHOR: Martin Bauer

PLACE OF BIRTH: Esslingen, Germany

DATE OF BIRTH: May 18, 1974

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

University of Oregon
Universität Stuttgart

DEGREES AWARDED:

Master of Science in Computer and Information Science, 1998,
University of Oregon
Bachelor of Science (Vordiplom) in Computer Science, 1996,
Universität Stuttgart

AREAS OF SPECIAL INTEREST:

Wearable Computing
User Interfaces
Artificial Intelligence

PROFESSIONAL EXPERIENCE:

Research Assistant, Department of Computer and Information Science,
University of Oregon, Eugene, 1998

Teaching Assistant, Department of Computer Science,
Universität Stuttgart, Germany, 1996

Research Assistant, Department of Computer Science,
Universität Stuttgart, Germany, 1996

Teaching Assistant, Department of Computer Science,
Universität Stuttgart, Germany, 1995

GRANTS:

German-American Fulbright Commission, 1997-98

PUBLICATIONS:

1. Bauer, Martin; Heiber, Timo; Kortuem, Gerd, and Segall, Zary. A Collaborative Wearable System with Remote Sensing. Proceedings of the Second International Symposium on Wearable Computers; 1998 Oct; Pittsburgh, PA. Los Alamitos, CA: IEEE Computer Society; 1998; c1998: pp.10-17.

2. Kortuem, Gerd; Segall, Zary, and Bauer, Martin. Context-Aware, Adaptive Wearable Computers as Remote Interfaces to 'Intelligent' Environments. Proceedings of the Second International Conference on Wearable Computers ; 1998; Pittsburgh, PA. Los Alamitos, CA: IEEE Computer Society; 1998; c1998: pp. 58-65.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Zary Segall, for his support. I am grateful to Gerd Kortuem for his substantial and continuous feedback that helped to improve the quality of this thesis considerably. Gerd and Jay Schneider assisted me in conducting the usability study for which I want to express my sincere thanks. I would also like to thank those who took the time to take part in this study.

I am grateful for the financial support of the German-American Fulbright Commission that made my studies here in the United States possible.

Finally, I want to thank the professors, the secretaries and everybody else who has made my stay at the University of Oregon a pleasant experience.

DEDICATION

To my parents and Vimala

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF GRAPHS

LIST OF FIGURES

Page

CHAPTER I

INTRODUCTION

How can wearable computing improve the collaboration of remote workers?

This question is becoming more and more important as companies are starting to employ wearable computers in their workplace. The technology of see-through head-mounted displays provides the basis for augmenting the physical reality as opposed to creating a virtual reality inside the computer. This allows us to build a system that supports real-world pointing by providing a remote user with the ability to control a pointer on the head-mounted display of a wearable computer. Thus, users can employ pointing as part of their conversation even though they are at different locations.

The central research question of this thesis is how effective and efficient remote pointing is in such a scenario. In order to address this question, we have formulated the following research hypotheses:

1. Users will prefer remote pointing to voice-only communication.
2. The combination of remote pointing and verbal communication is faster than just verbal communication. Remote pointing allows the users to focus quickly on the point of interest.

3. Users will utilize the option of freezing the live video image quite often to counterbalance the other user's head movement.
4. Our design mimics real world pointing. To support remote pointing, language will be used in the same way as in real-world situations.

In order to validate these hypotheses, we conducted a usability study on the basis of our prototype system. We had real users working with the system and they had to accomplish simple, given tasks. A detailed discussion of the hypotheses, the usability study and its results are presented in Chapter V.

Before going further into the details of the study and the prototype system, we need to give some more background information to provide a basis for the following discussion. The rest of this introduction gives an overview of the most important aspects, while Chapter II discusses issues in more depth and relates them to relevant previous work.

Wearable computing is a relatively new research area. As computer hardware has become smaller and at the same time more powerful in recent years, it is now possible to put a PC-like device into a case that can be worn on the body. Such a *wearable computer* is equipped with novel and advanced input and output devices like head-mounted displays (HMD), chording keyboards or speech recognition.

Figure 1 shows the author wearing a vest containing a wearable computer. A head-mounted display serves as an output device; the microphone and the chording keyboard

FIGURE 1. Wearable Computer Vest

are used as input devices. A more detailed description of our hardware is presented in Chapter IV.

The idea behind wearable computing is that the user is free to pursue other activities, but has computer support when he/she needs it. Oftentimes a wearable computer is used for providing automatic, context-sensitive and hands-free access to information.

Typical areas in which wearable computers are actually used include maintenance, repair, construction, manufacturing and on-the-job training. Some major companies have started utilizing wearable computers. For example, Boeing uses them in aircraft assembly [5] and the United Parcel Service for data collection [27].

In order to be able to assess the importance of remote pointing in a collaborative wearable setting, we have to look at the function of pointing in real-world scenarios.

Let's look at a typical example. Two hikers meet on an open trail somewhere in the mountains. One of them has a map, but is somewhat disoriented and doesn't know exactly where he/she is. The other hiker will look at the map and point at the approximate location of the meeting point. To verify it and to give the other person a better orientation, he/she will point out landmarks like peaks or lakes that are marked on the map and can be seen in reality. *Pointing out* in this context means both using the forefinger to point at an object and giving some additional verbal information for clarification.

With a wearable computer a user gets computer access in his/her natural physical environment - wherever he/she chooses to be. With a video conferencing application, other users can get a "window" into that environment. This facilitates computer-mediated collaboration on tasks involving objects in the physical environment of a wearable user [1].

Even though we use video-conferencing as a basis for our system, there are a number of characteristics involved that are different from most ordinary, desktop-based video-conferencing systems:

1. A video camera attached to the head-mounted display shows what the wearable user is seeing.

[1] We use the term *wearable user* for describing a user equipped with a wearable computer.

2. A low bandwidth, high delay wireless network link is used to connect local and remote users.
3. We introduce a remote pointer enabling a remote user to point at physical objects in the immediate view of the wearable user.

A traditional desktop-based video conferencing application offers video and voice communication, but doesn't give us the ability to point at real-world objects in our communication partner's environment.

A wearable computer on the other hand makes it possible to build a system that enables users to point at those real-world objects. Therefore, we call this form of pointing reality-based remote pointing, or just remote pointing.

At this point we need to elaborate a little bit more on how remote pointing actually works. Both the remote user and the wearable user have the video image captured by the wearable user's video camera on their displays. The position of the mouse cursor on the remote user's video image is transmitted over the network to the wearable computer. This position is then visualized as a remote cursor on the wearable user's video image as shown on the head-mounted display. Thereby, the remote user can point at any object in the current view of the wearable user. The wearable user just has to do a mapping from the video image to the real world.

The wearable user's part is illustrated in Figure 2. It shows the video camera that captures the image, the HMD that displays it and the remote pointer that is overlaid onto

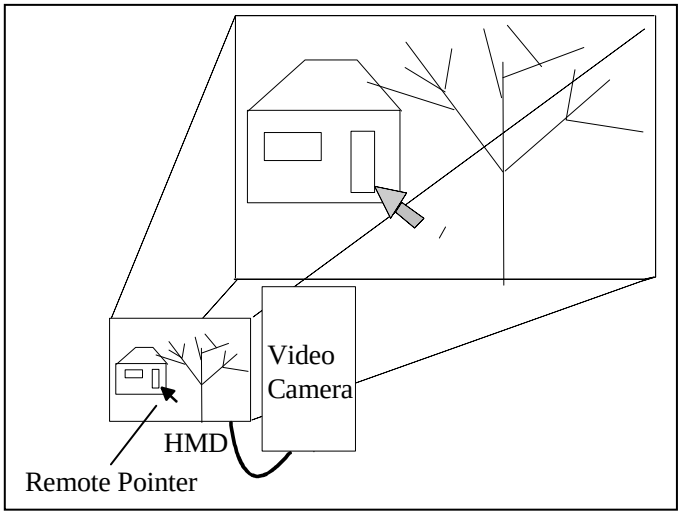


Figure 2. Remote Pointing

the video image. The grey pointer symbolizes the mapping of the remote pointer on the real object. So in this case, the remote user is pointing at the door of the house.

The wearable user's head movements present a problem for remote pointing because the video image is changing all the time. This makes it difficult to point at an object, since it changes its relative location on the video image. We dealt with this problem by giving the desktop user the ability to freeze the image. Then he/she can move the pointer on the still image.

We see pointing only as a first step for enhancing computer-mediated collaboration in wearable settings. A short discussion of possible other directions is given in Chapter II.

The system we built is actually a hybrid system as it involves both desktop and wearable computers. It allows highly mobile users with wearable computers to collaborate with office-based users at desktop computers. A typical example for this

would be the cooperation between mobile network technicians and experts, who primarily stay in their office. We discuss this scenario in Chapter III.

The rest of the thesis has the following structure: Chapter II provides a general motivation for remote pointing in a wearable setting and discusses other work that is relevant for this thesis. Chapter III describes the network maintenance scenario that served as motivation for the development of this system. Chapter IV presents the design and implementation of a prototype system. In Chapter V, as mentioned above, the usability study and its results are discussed. Finally, Chapter VI presents some conclusions.

CHAPTER II

MOTIVATION AND BACKGROUND

Wearable Computing

In this paragraph, we want to give an idea what wearable computing is. Instead of having your computer on a desk like a desktop computer, or carrying it around as a laptop computer, you wear the computer on your body, maybe even literally integrated into your clothing with signals running through special fibers. For using a wearable computer, input and output devices that differ from those of desktop or laptop computers are needed, e.g. head-mounted displays or one-handed keyboards. Maybe the most important difference to other forms of portable computing devices is that the computer is permanently accessible. The user doesn't have to take it out of a pocket or bag, open it and start it. – It can be running permanently, ready to be used.

One motivation for wearing a wearable computer is that you want to be able to walk around and do other things. This implies that using the computer may not be the exclusive task.

This research area is still relatively young and changing quickly. As a result the perception of what wearable computing is and what the research issues are varies greatly.

In [11] p.60 Kortuem et al. identify four elements as fundamental for the wearable computing idea:

1. Hands-free or one-handed operation
2. Mobility
3. Augmented Reality
4. Sensors and Perception

The prototype system that we present in this thesis covers all four elements. The wearable user does not need any input device for the remote pointing functionality. Therefore we have hands-free operation.

The wearable user can walk around and pursue all sorts of other physical activities. This means he/she is highly mobile. Though this mobility may be restricted to the area covered by the wireless network.

Video cameras and microphones are sensors that provide the user and the respective co-user with a perception of the environment. Finally, the system augments a wearable user's reality by providing a remote pointer. This pointer is pointing at objects in his/her real-world environment, thereby augmenting it.

Other researchers, e.g. at the Massachusetts Institute of Technology's Media Lab, have a different focus. In [17] p. 2 Steve Mann lists the following attribute in his definition:

Personal: Human and computer are inextricably intertwined.

In [24] p. 124 Bradley Rhodes emphasizes that wearable computers are *always on*.

A look at the work of Mann and Rhodes shows that they see a wearable computer as a form of personal extension that is always worn and always available. In this line of thought, a wearable system should bring an augmentation of human abilities, ultimately

forming a symbiosis. Therefore, the attributes listed above are very important for their work.

Our system, in contrast, uses wearable computers as a tool for remote collaboration. Its purpose is to facilitate the accomplishment of certain tasks. Whoever has to perform this task is going to wear the wearable computer and he/she will do so for the time required to accomplish the task. Therefore, in our scenario, the wearable computer is neither personal nor always on.

Related Research

Wearable Computing

In [25] Starner et al. describe an augmented reality system that allows, among other things, to overlay the environment with graphical images. *Augmented Reality* is quite often used in wearable computer systems. The reality is augmented with any sort of additional information ranging from simple text messages to complex 3D objects. In this example, a tag system is used to automatically identify an object and overlay the correct image that appears to be affixed to the object. These overlays can be associated with hyperlinks. As an example, where such a system may be useful, they mention a museum. It is often difficult to find the right amount of text describing an exhibit, so that people on average do not get bored, but interested visitors get enough information. With this overlaid hyperlink system, everybody can get the amount of information he/she wants.

Our system also augments reality. Instead of text messages or other objects, it provides a pointer. Since another user controls this pointer, we have a sort of collaborative augmented reality.

Collaborative Wearable Computing

In [2] Billinghurst et al. use virtual reality for a collaborative application, a wearable spatial conferencing space. Users are presented by pictures of themselves that are placed into the conferencing space. They can freely move around. The position of all participants in the virtual conferencing space can be seen on a small “radar-image” in the lower right corner of the HMD screen. A spatialized audio system gives the impression of being in the space. Participants can come closer together, form discussion groups or get together in bigger groups, thus modeling possible real world situations in a virtual space.

This system creates a virtual reality, a reality that exists inside a computer and has no relation to the physical world. The user becomes part of a virtual world and is acting in that virtual world. In a sense, we are doing the opposite. We give a remote user access to the physical world of another user. They are collaborating on tasks involving real world objects not computer generated ones.

In [14] Kraut et al. give an example of how a collaborative (wearable) computer system can be used for a maintenance task. They conducted a usability study in the field of bicycle repair. They studied the collaboration of a remote expert and a field worker in performing a set of repair tasks on a bicycle. Varying the communication technology,

they had either half duplex audio and video, full duplex audio and video, or full duplex audio only. In the video condition, the field worker's camera was pointing at the task, while the expert's camera showed the person. The field worker was wearing a helmet on which the HMD and the camera were mounted. Apart from the video, they had a shared application showing a bicycle repair manual. Their main interest was how the performance and communication differed, when using different kinds of communication technology. They also had a "control group" that had to perform the bicycle repair without any help from an expert. The study brought one surprising result:

> While having access to an expert dramatically improved performance, having better tools for communication with the expert did not improve the number of tasks completed, the average time per completed task or the performance quality. [14] p. 61

On the other hand they showed that even though the technology didn't change performance, it had a big influence on how people communicated, e.g. without a video connection, the field worker's descriptions of the current state played a much more important role.

This work is very much related to what we are doing. They have a similar scenario in which an expert collaborates with a mobile field worker. Among other things they compare communication with and without video. In contrast, we take audio and video communication as a basis and add remote pointing. Our goal however is very similar, as we want to find out, what effect a different technology has on communication. While they tested their system on a complex, real-world task that also involved other elements like the online-manual, we concentrate on the remote pointing aspect.

Desktop-Based Video Conferencing

Now that we have seen a wearable computer based videoconferencing system, we take a look at more traditional desktop computer based systems. Most such systems use the "talking heads" approach, e.g. [18] and [28]. This means, a video image showing the head and parts of a participant's upper body is transmitted to all (other) participants.

Such an approach is not feasible in a wearable setting. The wearable user is moving around. There is no place from which a camera could always show his/her head.

In [30] Whittaker reports that even in an office setting the real benefit of enhancing an audio connection with a live video image showing the head of the person is limited. He only finds the following evidence:

> The outcome of affect-dependent tasks such as conflict resolution or person perception is different when video information is present, compared with audio only communication. [30] p. 295

We have a different situation, when video is used as data. "Video-as-data"[30] means that the camera is pointing away from the participant and onto an object of interest.

Gaver et al. [6] developed a video conferencing system that allows the user to choose from different views. In addition to the described face-to-face view, there were cameras pointing at the main workspace and onto a space on the desk to allow document sharing. In some cases, there was a camera that showed the whole room. They recorded the times the users spent in each of the different views. The face-to-face view was only used 11% of the time, while the views showing part of the workplace were used 47% of the time.

This leads Whittaker to the following conclusion:

> Information about gaze and gesture of the other conversational participant seems to be less important than information about the object itself.
> [30] p. 301

Therefore it seems to be reasonable to use such a camera setting for remote collaboration involving wearable users.

In [15], Kuzuoka et al. describe the SharedView system, a system that led to the GestureCam system, which is the main focus of the paper. The SharedView resembles our system in many ways. It is used in an instructor – operator setting, where the instructor tells the operator how to operate a Machining Center. The operator is wearing a head-mounted display and a camera is pointing in the direction of view. The image of the camera is shown on a display on the instructor's site. Another camera is pointing on that display and the instructor can use any pointing gestures with his hand on his/her display. This camera image is then transmitted back to the HMD of the operator, so the operator sees the image of his own camera augmented by the real gestures of the instructor.

In this case, a video image is transmitted over the network in two directions. This does not present a problem, if network resources are plentiful, but would be a severe problem in the case of a wireless connection that may already present a bottleneck in the case of a one-way connection. A wearable system requires such a connection, so the approach doesn't seem feasible. The advantage of the system is that the instructor can use natural gestures for pointing that can be more expressive than simple cursors. The disadvantage is that the instructor, the camera and the display have to be positioned in a

way that allows the operator to see the important aspects of the picture. This means that the flexibility and mobility of the instructor is somewhat restricted.

The GestureCam system itself relies on remotely controlled cameras that are mounted on camera actuators. A construction like this would be impractical for a wearable scenario, so we won't discuss it here.

Telepointing

Telepointers are used in a lot of groupware applications, e.g. MMConf [3], CoLab [26] or GroupWeb[7]. A telepointer is a mouse pointer controlled by a remote participant and is used for pointing at objects in shared windows. Systems differ in the number of telepointers and in the way they are controlled. MMConf provides an infrastructure for building shared multimedia applications. It has only one shared group pointer that can be invoked only by the currently active user. The active user is determined by a token that is passed around among the participants. CoLab, a computer-based meeting-room system has multiple cursors, but they are only displayed on request, whereas GroupWeb, a shared WWW Browser, has multiple cursors that are shown constantly.

The difference between the telepointers in desktop-based applications like the ones mentioned in the previous paragraph and our remote pointer is that the telepointers point at shared virtual objects, whereas our remote pointer points at real-world objects in the physical space of a remote user. Because of this semantic difference we chose to use the expression *reality-based remote pointer* or just *remote pointer* instead of *telepointer*.

After explaining why we distinguish between telepointing and remote pointing, we need to take a closer look at our motivation for remote pointing: “real-world” pointing. As already mentioned in the introduction, the gesture of pointing is an important element of communication in real-world scenarios.

We have seen the example of the two hikers in the mountains. Another example is an architect and a civil engineer standing at a construction site. They are talking about the progress of the construction work. While doing that, they point out locations that, for various reasons, need some attention.

Other simple examples would be pointing at a flower on the wayside or pointing out a person you have spotted in a crowd of people. Children in particular use a lot of pointing, e.g. for telling their mother who has hurt them.

Pointing also has some social implications. It would be considered rude to openly point at a person nearby. But as this is not relevant for our purpose, we will not discuss it here.

Now, why do we need to point? Imagine trying to achieve the same result as in the given examples without being able to point -especially in a situation in which something is hard to spot or not obvious. It would be much more difficult to put the same information into words. The reason is that we often have a complex reality that is hard to describe. In order to focus on the point we are really interested in, we would have to give a lot of extra information. Pointing gives us the necessary focus.

An important characteristic of (real-world) pointing is that it can only be used in face-to-face situations. It is vital for the person pointing and the person observing to be physically close. Then they can see the same objects and the observer can figure out, what is being pointed at. They can also talk to make sure that they understand each other.

In a computer-mediated communication, people are usually far away from each other. Otherwise they wouldn't need the computers in the first place. Still, we would like to have something similar to pointing in order to simplify communication.

Let's go back to the example with the civil engineer and the architect. Imagine that only the civil engineer actually is at the construction site and the architect is sitting in his office. If they have a videoconferencing system with a camera pointing at the construction site, they can still talk about the progress of the work.

The important thing that is missing when comparing the two scenarios is the ability to point.

Remote pointing offers exactly this ability. It is our solution to the problem of porting pointing into the realm of computer-mediated communication.

There might even be an advantage of remote pointing over real-world pointing. The wearable user sees precisely what the desktop user is pointing at. The pointer is on the object itself. In real world pointing, people can often give only the direction. Depending on how close people are to each other physically, the viewing angle can be significantly different. Remote Pointing is done on a 2D-projection, so we can point at the object

directly. The disadvantage is that we have to do the mapping from the projection onto the real world.

Beyond Remote Pointing

After introducing remote pointing, an obvious question is if there are other ways to enhance remote collaboration. In [1] p. 13 Bauer et al. identified a number of collaboration primitives from which a collaborative system can be assembled:

1. *Remote awareness* provides some form of presentation of each participant, so people are aware of who is involved in the collaboration.
2. *Remote presence* goes a step further than remote awareness. It provides some form of live representation of the participants, e.g. live video or a 3D avatar.
3. *Remote presentation* allows a user to superimpose images over the remote users' real-world view.
4. *Remote pointing* needs no further explanation at this point.
5. *Remote sensing* means that a user has direct access to information from sensors that are connected to a remote wearable computer.
6. *Remote manipulation* allows a user to manipulate objects in the environment of a remote wearable user.

So the focus of this thesis, remote pointing, is one of these collaboration primitives. It enriches collaboration by providing a pointer into a physical environment. Ultimately, we want to go beyond that.

In [1] a collaborative system that supports remote sensing is described. It includes sensors for location, object identity and, as it is also based on the network maintenance scenario that we describe in the next chapter, a packet sniffer for analyzing network traffic.

Summary

Videoconferencing provides the basis for computer-mediated collaboration in a real-world environment. As we have shown, it is widely used in different scenarios. In the context of wearable computing, it is natural to have the camera pointing away from the user onto the area of interest, which is called "video-as-data" in the literature. [14] gives an example of a videoconferencing system in a wearable domain. Typical characteristics for a wearable system are the mobility of the user and the hands-free or one-handed operation of the computer.

The importance of pointing for the communication in real-world scenarios motivated us to port it into the domain of computer-mediated collaboration in wearable settings. We did so by implementing a remote pointer that is overlaid over a video image showing the view of the physical world as seen by the wearable user. The video image is displayed on a head-mounted display, so the user can see image and reality at the same time. We thereby augment the reality of the wearable user.

Telepointers are used in desktop-based collaborative applications for pointing at virtual objects that only exist inside the computer. To distinguish our pointers that are

used to point at objects in a physical environment from those telepointers, we called them *remote pointers*.

In order to find out, if remote pointing is as effective and efficient as we would expect a portation of real-world pointing to be, we conducted a usability study that is presented in Chapter V.

Even though the idea of using remote pointing[2] in collaborative wearable systems existed and may well have been implemented, nobody has evaluated the use of remote pointing in a wearable setting by conducting a usability study. The use of image freezing in that context seems natural, although we haven't seen it mentioned anywhere. We therefore think that this thesis makes a valuable contribution to the field.

[2] After showing remote pointing as part of our presentation of [1] at the Second International Symposium on Wearable Computing, Steve Mann mentioned that he had used remote cursors in some of his "early work". We couldn't find any reference in the papers we have, except for a side remark in [16], describing the idea of using a remote cursor while shopping. An email to Steve Mann asking for more information has not been answered.

CHAPTER III

SCENARIO

This thesis grew out of the NETMAN project [12] that is being developed by the Wearable Research Group at the University of Oregon. Although the system developed for this thesis handles only a few aspects of the envisioned NETMAN system, we will give a short description of the underlying scenario, as it had an influence on the development of our system and ultimately on the artificial task of our usability study.

The NETMAN system is designed as a collaborative system for wearable and desktop computers that supports technicians maintaining a computer network on a university campus. (Figure 3) The requirements for this project were collected in close cooperation with the Computing Center of the University of Oregon, which is responsible for maintaining the computer and network installations throughout the University campus.

The Computing Center has two different kinds of employees. There are a few full-time employees who are experts in their fields. They spend most of their time in their office and don't perform any routine repairs. Then there are a number of less experienced *field-technicians*, often students, who are fairly literate in using computers. They also have some limited knowledge of computer maintenance and repair, which is

usually sufficient for solving routine problems. Typical tasks are installation of new network equipment, regularly scheduled inspections, trouble shooting of network faults and repair/replacement of faulty equipment.

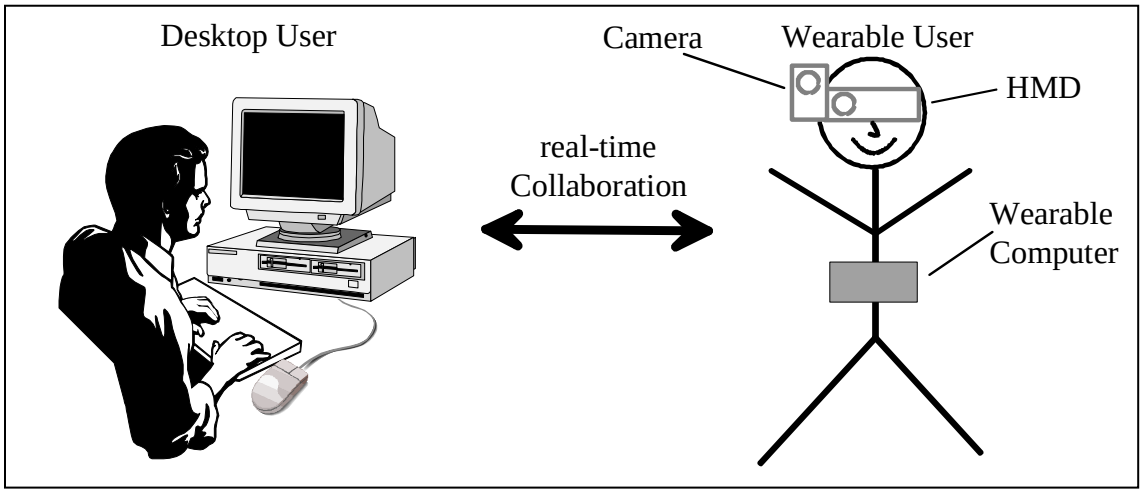


FIGURE 3.1: Collaboration Between Desktop and Wearable User.

The field-technicians usually carry a lot of equipment, primarily for communication purposes, such as cellular phones, walkie-talkies, pagers, and in some cases notebook computers. Quite often, when they have discovered a problem, they need some more information to fix it. This can involve calling somebody on the phone, going back to the office to talk to an expert in person, or looking up information that is mostly available in the form of online manuals. Thus a good deal of time is spent going back and forth between the site of the problem and the office.

Altogether, the current practice is inefficient in terms of the necessary equipment and the time spent on the task.

To remedy those shortcomings mobile computer support is suggested. A computer can replace the whole range of communication devices that field-technicians currently carry around. It provides instant access to online documentation and to communication software, from email to videoconferencing. Using a wearable design for such a system has a number of advantages. First, the field-technicians have to be highly mobile, since they may have to change location several times to find a problem. Second, they have to do physical work, like opening computer cases. In this kind of scenario, laptop computers would be impractical. They have to be taken out of a bag, placed on a table and turned on. A wearable computer, however, can be running the whole time, is always accessible and is not in the way when performing physical tasks.

A further advantage is that communication can be augmented. There are a lot of cases in which a phone conversation between a field worker and an expert is insufficient. The expert does not see what the field worker is seeing and it is often quite difficult to convey information by verbal description only. Adding a video camera that is pointing the way the technician is looking and using a videoconferencing system can provide crucial information that makes communication far more efficient.

However, there are still questions that are difficult to answer. The following examples are typical for this type of questions: Where do I have to plug in this cable? Where should this connect to? Which card do I need to remove?

For all these cases, it would be nice to be able to point at the right location, since it is far easier than putting the same information into words.

This is where the idea of having a remote pointer comes in. If the wearable user is equipped with a head-mounted display and his/her own video image is shown on the screen, a pointer can be overlaid that is controlled by the remote user in the office. Thereby the remote user can answer the questions mentioned above. For example he/she could say, “Plug the cable in here.”, and point at the correct location. All the wearable user has to do is to map from the video image onto the reality and he/she can plug in the cable.

As this kind of conversation is typical for the network maintenance domain, remote pointing could play an important role. Yet, remote pointing had not been studied in this kind of setting. Therefore it was unclear, if remote pointing could really be as helpful for the communication as the previous example suggested.

For this reason, we decided to evaluate remote pointing. The next chapter discusses the prototype system we built for that purpose and the following chapter presents a usability study that examines the role of remote pointing for communication in a similar setting.

CHAPTER IV

SYSTEM DESCRIPTION

This Chapter describes the prototype system we actually implemented. It comprises the collaboration component of the NETMAN system that was outlined in the previous chapter.

We first give a description of the system design identifying the different components, the role of the users and their interaction. Then we discuss our wearable hardware. We present a survey of videoconferencing solutions with toolkits that served as a basis for our decision regarding the foundation on which we wanted to build our system. After that, the structure of Microsoft NetMeeting, our eventual choice, is shown. We conclude by discussing the user interface of the implemented prototype system.

Design

In our scenario we have an office-based user working at a desktop computer collaborating with a user in the field carrying a wearable computer. The wearable user has a head-mounted display (HMD) to which a microphone, a headset and a video camera are attached. The desktop user also has a microphone and speakers, and the two

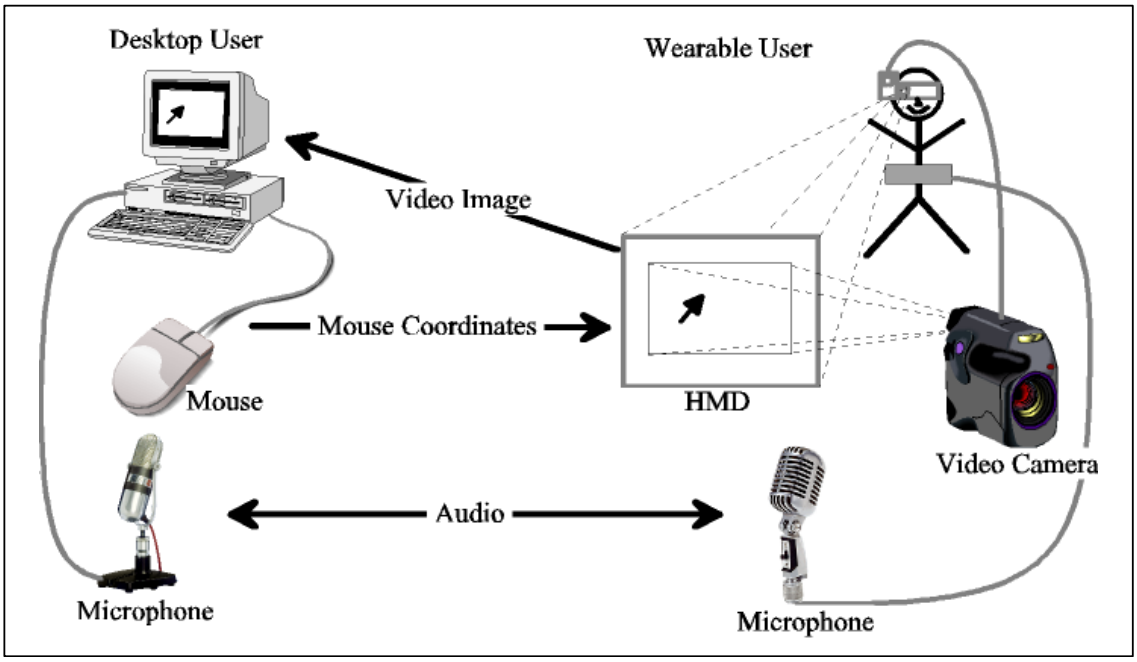


FIGURE 4. System Overview

can communicate via an audio connection. (Figure 4) The image of the video camera is displayed on both the HMD and the monitor of the desktop computer.

Remote pointing is implemented in the following way: When the desktop user moves his mouse onto the video image, a second cursor appears on the wearable user's video image. This pointer represents the desktop user's mouse. By pressing his/her mouse button the desktop user can freeze both video images, his/her own and the one on the wearable user's HMD. Then he/she can freely move the mouse pointer on the still video image to point at the chosen object. A small second window shows the live video stream even when the main window is frozen.

The video camera points at the workspace of the wearable user, in the same direction as his/her eyes. Therefore, the video image gives the desktop user a window

into the wearable user's physical environment. From his/her perspective it will be the actual object he/she is pointing at, not the point on the screen. From the wearable user's point of view, the video image shows what he is looking at in the real world. There should be an easy mapping between the objects on the video image and the real-world objects.

Controlling the remote pointer with the mouse should be intuitive since most computers today have a graphical user interface that uses a mouse as an input device. Users are familiar with moving a mouse cursor across the screen, which is exactly what the desktop user in our system has to do. Therefore, access to remote pointing is fast and easy - no further training is necessary. The wearable user has a remote cursor on the screen showing the position at which the desktop user is pointing.

Image freezing allows the desktop user to control the wearable user's image. This is important for accurate pointing, since the wearable user may constantly move his head. While the image is frozen, the desktop user can observe the wearable user's action in the additional small window that still shows the continuous video stream, so incorrect actions can be corrected immediately.

Wearable Hardware

In 1996, the Wearable Research Group at the University of Oregon built its own wearable computer from commercial off-the-shelf components on the based on a Texas Instruments motherboard with a Pentium 75. For more information about that computer see [6]. Now, we mainly use a commercial product, a ViA Wearable™. It offers more

interfaces for the use of peripheral devices, which is especially useful when developing and testing applications.

FIGURE 5. ViA Wearable Computer (Case, Display and Battery)

ViA Wearable Computer

The ViA Wearable™ (Figure 5) is based on an AMD 586 processor with a clock frequency of 133 MHz. It comes with a main memory of 24 MB DRAM. Three connected pods contain the computer hardware. The CPU and primary system components are housed in the center pod, while the other two pods offer four PCMCIA Card slots and indicators, switches and connectors to the outside.

The I/O connector either connects to a hand-held display with pen input or to a standard I/O cable which in turn has connectors for a keyboard, a mouse, stereo speakers, a microphone, a VGA compatible display and two serial ports.

The built-in video graphics adapter (VGA) supports a display resolution of 640x480 with 256 colors on the hand-held active-matrix LCD display and a resolution of 800x600 on other, Super VGA compatible displays, e.g. on a normal desktop monitor.

The hand-held display contains a touch panel that allows user input by means of a pen. A software-based on-screen keyboard can be used for text input.

One designated PCMCIA card slot is reserved for the PCMCIA Type III compatible system hard disk that has 340 MB of memory. We use the other slots in the following way: We added an additional hard disk with a capacity of 260 MB. A video capture card allows us to connect a video camera. During software development we use an Ethernet card to connect to the Internet and our Windows NT based PC network.

Further, we have a modem card, a card for a parallel port and for a serial port, which is needed if we use the LCD display instead of the standard I/O cable.

One or two batteries power the wearable computer. Having two different power connections makes it possible to switch batteries in operation. We just need to replace one of them, while the computer is running on the other. Since the batteries we use don’t last very long (usually not more than one or two hours), this is a very important feature.

The wearable computer comes with a pre-installed version of Windows 95 that uses special drivers to support optimally its special hardware.

More information about the ViA wearable computer can be found in [29].

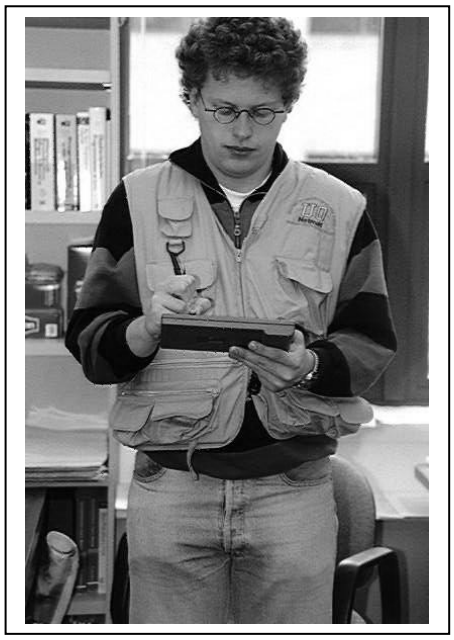


FIGURE 6. Wearable Vest

The whole computer can be worn around the waist using a belt. Alternatively, we use a specially designed vest. (Figure 6) The wearable computer is placed in a pouch on the back of the vest. The batteries counterbalance the weight in the front pouches, so the vest has a comfortable fit. The cables to the batteries and the input and output devices run inside the vest.

Head-Mounted Display

In our opinion, one of the most important features of wearable computing is that it is possible to operate it hands-free or at least one-handed. Therefore, the hand-held display offered by the ViA wearable computer may not be suitable for some applications, as it doesn't allow a one-handed mode of operation.

FIGURE 7. i-glasses!™

Instead, we use a head-mounted display (HMD). A HMD consists of a sort of goggles with one or two eyeglasses onto which a video image is projected. In Figure 7 you can see our HMD, i-glasses!™ from Virtual i.O, Inc. They are easily available and relatively inexpensive. In terms of optical quality much better products are available. They are not necessarily intended for wearable computing and are mainly used for viewing videotapes, laserdiscs and TV, or for playing electronic games. Their 3D and head-tracking capability can be used to create a virtual reality environment.

The i-glasses!™ can be operated in "see through" mode, which allows the user to maintain the environment in the field of vision, or in "immersive" mode, where this is no longer possible. This is achieved by adding a clip-on visor.

For our purpose, the user needs to act in the real world environment; therefore, we operate in "see through" mode. In addition to that, we have removed one of the eyeglasses as this allows an unobstructed and more natural view. The user can "switch" mentally between the screen and the real world environment.

A special PC adapter kit is necessary to convert the VGA into a video signal. The input signal has to be 640x480 pixel, between 60 and 70 Hz and without any stretching of the video image, which is done by some Laptop Computers to improve the utilized space on their LCD display. The adapter needs a separate power supply, which is realized through an AC Power converter. This makes it difficult to make the i-glasses!™ truly "wearable".

An eyeglass is realized through a 0.7 inch Liquid Crystal Display. It has a resolution of 180,000 Pixel that is split between the three colors. So, even though we get a VGA signal as input, the resolution is more like the real resolution is 262x230 and feels more like an old CGA (Color Graphics Adapter) display. An improved version with a resolution of 360.000 Pixel is now available. A single headphone is incorporated into the HMD frame.

For more information about i-glasses!™ confer [9].

Microphone and Camera

In order to allow voice communication, we added a microphone to the HMD. The attachment of a small camera allows us to transmit the wearable user's view to a co-worker (Figure 8). In addition, we can display it on the user's own HMD, overlaying it with additional contextual information and thereby creating an augmented reality environment.

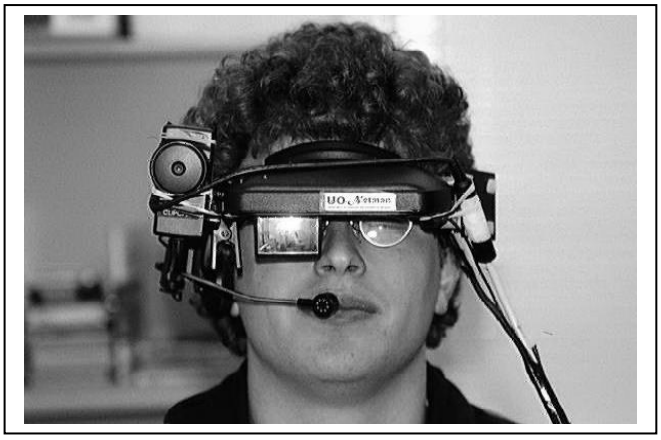


FIGURE 8. Author with HMD, Camera and Microphone

We use a Nogatech ClipCam, a small color NTSC video camera. The camera comes with a PCMCIA video capture card. The lightweight Camera is powered by the computer it is attached to.

Unfortunately it only has a manual focus, but as the working distance to the objects in the user's view usually doesn't vary a lot, it is no major problem in practice. The second drawback is that currently only drivers for Windows 95 are available, nothing for Windows NT. Our wearable is running Windows 95, so it is not quite as important.

For more information about the camera see [23].

Chording Keyboard

As the user of a wearable computer should be able to move around freely having at least one free hand, we cannot use standard input devices such as mouse or keyboard. Ideally, we would like to use voice input. Voice recognition software has made some

progress, so we could probably use it for entering a restricted set of commands. Free text input is far more critical. We also have to keep in mind that the wearable computer might be used in places with a lot of background noise. In addition, we use speech as a means of communication between co-workers, which might interfere with the voice recognition.

Therefore, alternative input devices have to be considered. We use a commercial chorded keyboard called Twiddler™ that is produced by Handykey Corporation. (Figure 9) It is designed for use with one hand, which can be either the left or the right hand. It performs the functions of both mouse and computer keyboard. The twelve keys on the front are ordered in a grid of four rows and three columns and are operated by the four fingers; the six keys on the top are pressed with the thumb. All the functions of a full 101-key keyboard can be performed. Each single key on the front side produces a character when letting go of the key. To get all the other characters available on an ordinary keyboard, you have to press a combination of two or more keys. Macros can be assigned to unused combinations.

The main problem of using a chorded keyboard like the Twiddler™ is that the learning curve is very steep and most potential users might not want to spend the learning time necessary for achieving a reasonable typing speed.

More information can be found in [8].

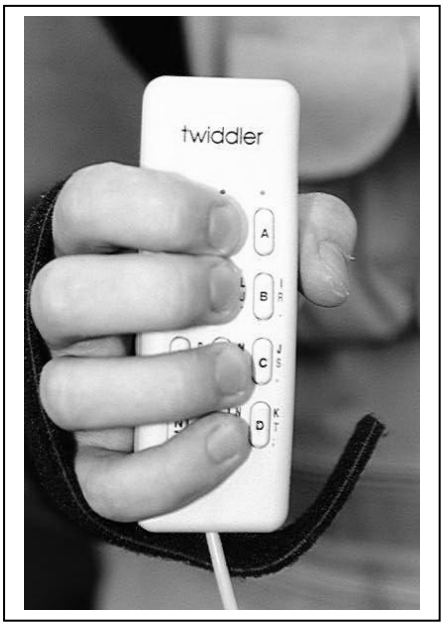


FIGURE 9 Twiddler™

Wireless Modem

Since we want to use the wearable computer in a collaborative setting, we need some form of wireless communication to connect to the network. We use a Ricochet wireless modem produced by Metricom Incorporation. Metricom provides a wireless network that covers the whole University of Oregon campus. It operates on radio frequencies in the 902-928 MHz band.

The modem is attached to a serial port and offers a speed of about 28.8 kilobytes per second. It uses its own rechargeable battery as a power supply, but can also be operated using a net adapter.

The relatively small communication bandwidth is a problem when transferring a live video stream. Getting a connection can also be a problem, especially inside buildings. This may be due to wires or other metal used in walls that block the reception.

For more information see [19].

Software

Our goal was to develop a system that supports video and audio conferencing and allows us to overlay a remote pointer on the video image. We wanted to integrate the components in our own application with its own user interface. As we were not interested in developing a totally new video conferencing system, we looked for a solution that provided us with a basis on which to build our application. Essentially it had to be something that could handle live transmission of video, audio and data over the network. For this prototype we needed the data transmission only to transmit the coordinates of the remote pointer.

In addition, there were a few more points to take into consideration. We needed something that allowed us to receive a useful video image, even in case of a very limited communication bandwidth. This is the case when we are using the Ricochet wireless modems that provide only 28.8 kilobytes per second.

For our current solution, we only have one-to-one communication between a wearable computer and a desktop PC. For future versions, it might be interesting to have more than two people collaborating, maybe even multiple people with wearable computers. So at least it is interesting to know if the solution is extensible.

Ideally, we wanted a system that was freely available and runs under Microsoft's Windows 95 operating system. As we are mainly developing on Windows NT machines, support for NT would be desirable, too.

So we looked around for products that seemed promising. We give a very short discussion of what we've considered. This should give an idea of why we came up with Microsoft NetMeeting as a basis for the communication part of our prototype.

Survey of Video Conferencing Solutions

Java Media Framework

Java is a good basis for developing wearable applications and has the advantage of being a platform independent solution. We therefore looked at the Java™ Media Framework API (JMF) that is being developed by Sun Microsystems, Silicon Graphics Inc. and Intel Corporation. It specifies a unified architecture, messaging protocol, and programming interface for media players, media capture, and conferencing. Version 1.0 supports synchronization, control, processing, and presentation of compressed streaming and stored time-based media including video and audio. Unfortunately there is no standard for capturing of live video yet. Without support for video capturing we could not use it for our project.

CU-SeeMe

CU-SeeMe is a free videoconferencing program that was developed at Cornell University. It can be used to set up videoconferences between PC or Macintosh

computers that are connected to the Internet. “Reflector” software can be used to enable multiparty videoconferencing.

Unfortunately CU-SeeMe is a standalone product that we can’t easily integrate into our own application and user interface.

Microsoft NetShow

Microsoft NetShow allows streaming of audio and video over the network. The drawback is that it needs Windows NT Server and it doesn’t handle data. As our wearable computer needs Windows 95 and we wanted to handle audio, video and data in a uniform way, this didn’t really fit our needs.

Microsoft NetMeeting

Microsoft NetMeeting is a conferencing application. It comes with a user interface that combines access to audio/video conferencing and application sharing. File Transfer, Chat and Whiteboard applications are also provided. NetMeeting works with any video capture card or camera that supports Video for Windows and runs under Windows 95 as well as under Windows NT.

Microsoft offers a software development kit (SDK) that allows people to “incorporate conferencing technology into a variety of applications using C, C++ or Visual Basic.”

The Microsoft NetMeeting™ SDK, as well as NetMeeting 2.1, can be downloaded free of charge from the Microsoft NetMeeting web site. [20]

As it turned out, NetMeeting was the only product that fulfilled our requirements for providing a basis for our system. Further down, we'll show that it is by no means ideal, but at the time we developed our prototype, we didn't have much of a choice.

Programming Environment

Choosing NetMeeting already narrowed down the choice of possible programming environments. Basically, Microsoft only supports Visual C++ and via ActiveX control Visual Basic as an interface to NetMeeting. It is possible that J/Direct, a feature of the Microsoft Virtual Machine that allows the call of WIN32API functions from Java, would have allowed us to use J++, the Microsoft version of Java. But this would have added another layer and the main advantage of Java, its platform independence, would have been lost anyway, so we didn't pursue this direction.

Instead we decided to use Microsoft Visual C++ and the Microsoft Foundation Classes (MFC).

The Microsoft Foundation classes implement an application framework. Visual C++ has a Class Wizard that uses that framework to produce a program skeleton into which the programmer can fit the actual application code.

The Microsoft Foundation Classes also provide reusable code in the form of abstractions of commonly applied programming idioms that are useful, when programming Windows applications.

The main drawback of using NetMeeting was the poor documentation that Microsoft provided with its SDK. While it was complete in terms of interface

descriptions, it didn't provide a lot of insight into the complicated relationship between the objects involved. Instead, we had to rely on undocumented sample code.

Recently, Alex Krassel and Steve Robinson from Panther Software have provided a workshop tutorial on the Web: "Microsoft's NetMeeting 2.1 COM Interfaces: Understanding how they work" [13] that gives a good overview on how NetMeeting works.

NetMeeting

In order to start a NetMeeting conference, we first need to place a call to a user on another computer. If the other user is running NetMeeting he/she will be asked to join the conference. If he/she accepts, a conference is set up. It is up to the participants to decide, if they want to send audio or video, or if they want to enable application sharing. NetMeeting creates a channel for every type of data it handles. As we'll see in the next section, a different protocol is used for each of them. We have one audio channel, one video channel, one channel for file transfer etc. Data channels are a little bit different, because there can be any number of them in one application. If participants in a conference want to communicate on a data channel, they each have to know the globally unique identifier that was assigned to that particular data channel.

Underlying NetMeeting Architecture

Microsoft uses a layered architecture for NetMeeting shown in Fig. 10, which is a simplified version of the diagram that can be found in Microsoft's online documentation. [22]

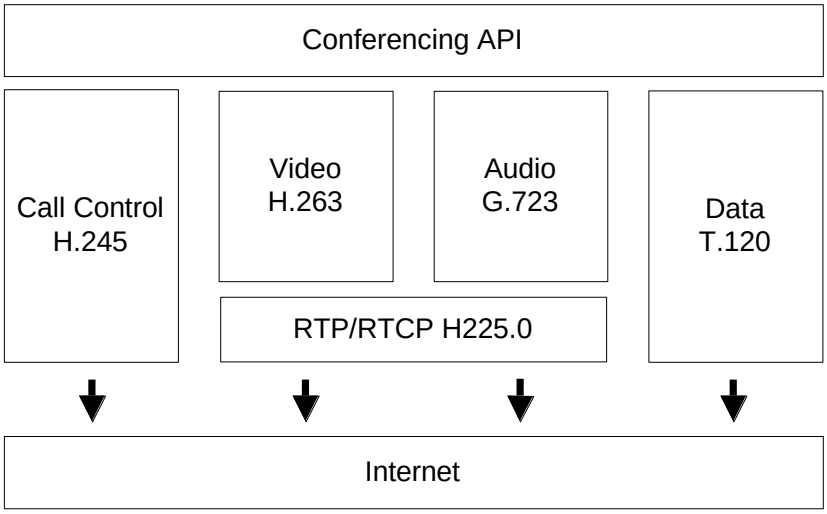


FIGURE 10. NetMeeting Architecture

Under the conferencing API layer that we will discuss in the next section, Microsoft uses a number of components, each handling a different type of data using a standardized protocol. The call control sets up a NetMeeting call using the H.245 standard for call control.

The video and audio streams are compressed on the sender side, transported over the Internet, and decompressed at the receiver side using H.263 and G.723 protocols respectively. The Real time Transport Protocol (RTP) handles the actual transport. The T.120 protocol is used for data conferencing

Different types of data have different characteristics. Therefore it makes sense to use different protocols for their transmission. Video and audio data have to arrive at the right time, otherwise the data becomes useless. On the other hand, it doesn't matter much, if a small piece of information is missing. Data and control messages on the other hand may not be as sensitive as far as the timing is concerned, but they mustn't get lost on the way. The different protocols take care of these requirements.

NetMeeting API

The NetMeeting Software Development Kit (SDK) provides us with a Component Object Model (COM) Interface to the NetMeeting functionality.

The Common Object Model from Microsoft forms a foundation for higher-level software services. It gives us a binary standard for component interoperability that is programming language independent. Functions can be grouped into interfaces. Objects implement these interfaces. The corresponding functions can then be called from other objects.

Every COM object has to provide a basic interface for dynamic discovery of interfaces and reference counting. Reference counting is used to track the lifetime of an object, so it can delete itself when appropriate.

More Information about the Component Object Model can be found in [4]

NetMeeting uses COM Interfaces for communication between objects. The following section shows how the interfaces and the objects that implement them form a structure that underlies our system.

Basically, there are three different levels in the NetMeeting Object hierarchy. They can be classified as Conference Manager level, Call/Conference level and Channel level. The difference between a call and a conference is that “a Conference object has channels and participants (members) while a Call object refers to a specific computer or user. A call is an invitation to join a conference.“ (Microsoft NetMeeting FAQ[21]).

	Client Classes	Server Classes
Manager Level	Manager_Notify_Sink implements INmManagerNotify Interface	Conference_Manager implements INmManager Interface
Call/ Conference Level	Call_Notify_Sink implements INmCallNotify Interface	Call implements INmCall Interface
	Conference_Notify_Sink implements INmConferenceNotify	Conference implements INmConference
Channel Level	Channel_Notify_Sink implements INmChannelNotify	Channel implements INmChannel

FIGURE 11. NetMeeting Object Levels

Figure 11 shows the three levels. Each class has to implement a certain interface, so the object instances can communicate with each other. NetMeeting provides the classes on the server side. On initialization it will create a Conference Manager Object.

The classes on the client side have to be written by the application programmer. Before initializing NetMeeting, a notify sink object has to be created. It will be bound to

the server object on initialization, so the server can notify the client side in case of an event, e.g. the creation of a conference.

Figure 12 shows the NetMeeting objects that are relevant for our system and how they interact, when the system is running. As mentioned above, we lacked a decent description of the NetMeeting COM interfaces, when we started with our implementation. Therefore, our implementation closely follows two sample programs that were supplied by Microsoft as part of the SDK: AVPhone, a video phone application and NMChat, a chat application that uses a data channel.

The following gives a detailed description of the implementation:

All information needed for managing conferences and calls on the client side – that means the first two levels of the hierarchy - is wrapped in a *Conf* object.

On initialization of the Conference Manager, the types of channel notifications that are allowed have to be defined: audio, video, data, file transfer, application sharing or any combination of these, including the option of getting no notifications at all. NetMeeting also needs a reference to our Manager Notify Sink object, so the client can be notified, e.g. when a new call or conference has been created.

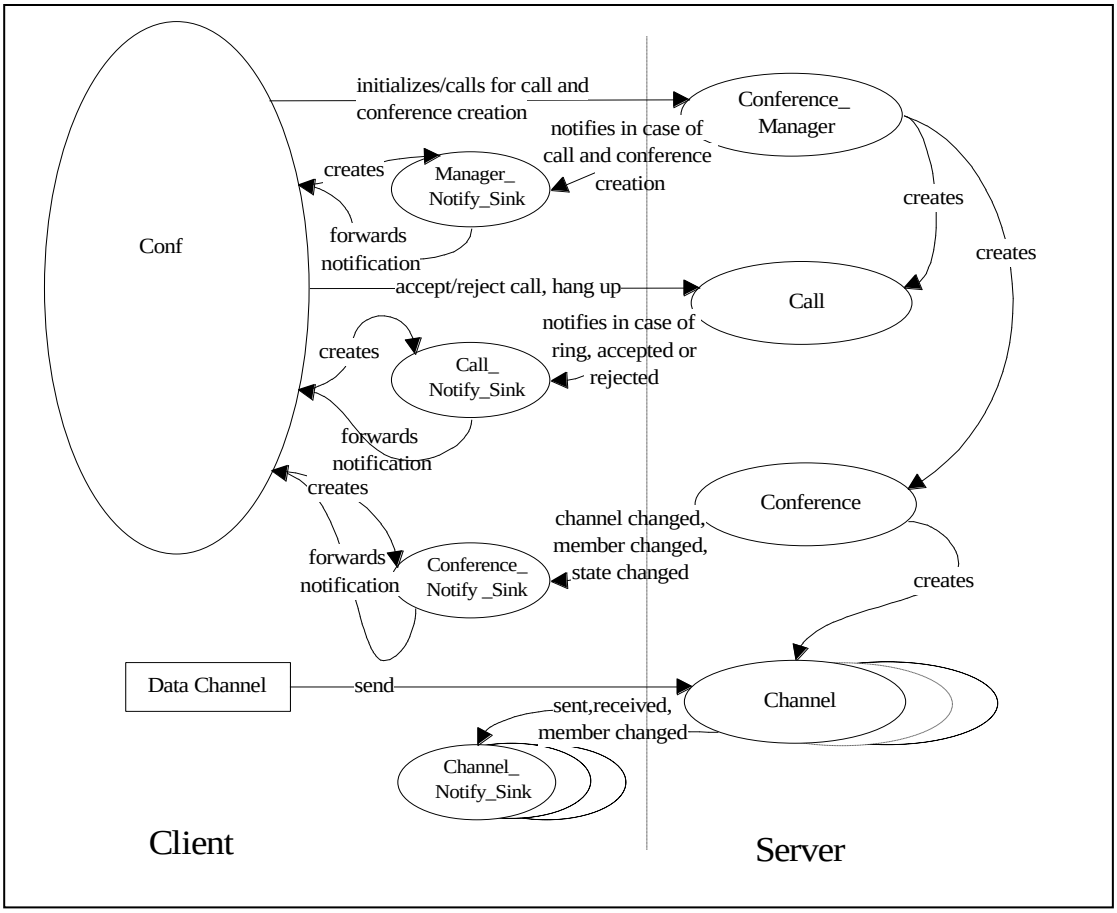


FIGURE 12. NetMeeting Objects in Our System

Both conference and call creation can either be initiated by the local machine or by another machine. In case of a call, it can either be accepted or rejected. If there is no reaction, a timeout is used to reject the call.

When a call is initiated, the conference manager creates the call object and notifies the Manager Notify Sink object. When a call is successfully established, the Conference Manager will create a conference object and again notify the Manager Notify Sink object.

On instantiation of the Conference Object, audio, video, file transfer and application sharing channels are automatically created. As mentioned before, data channels are a little bit different and therefore they have to be created “manually”.

The Conference Notify Sink object gets notified, if there is a change in channels, members or in the state of the conference. A channel change is reported, when a new channel is added, removed or updated. The different events can be distinguished through a notify flag.

NetMeeting doesn’t give direct access to the video stream. The application can only change a few parameters like image quality, size, position and visibility.

A data channel is the only kind of channel that has to be explicitly created by the application.

In our system, a data channel is used to transmit the coordinates of the remote pointer. It is hooked to a global variable, so it can be accessed from anywhere in the program.

A data channel is handled very much like the calls and conferences described above. The Channel Notify Sink object gets notified in the event of a member change, if data has been sent or data has been received.

Details on our program structure, e.g. how the discussed objects are integrated in our program, can be found in Appendix A.

User Interface

After starting the program, we first have to set up a conference. To initiate a NetMeeting conference, we have to place a call to another computer. Therefore we select the ‘Call’ option from the ‘NetMeeting’ menu (Figure 13). A dialog box pops up, where we can enter the name of the computer we want to call (Figure 14). By clicking on the ‘Call’ button, we place a NetMeeting call.

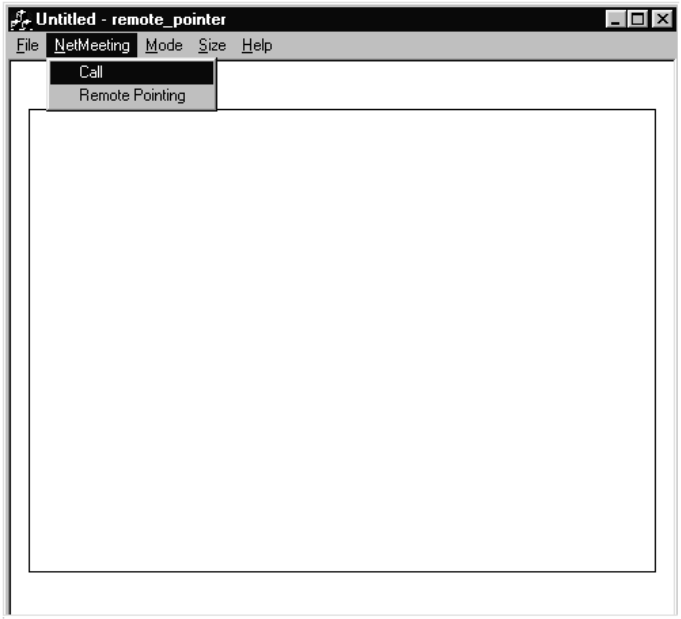


FIGURE 13. NetMeeting Menu

If the other user accepts the call, a conference is initialized and the video image from the wearable computer will be shown on both displays (Figure 4.11).

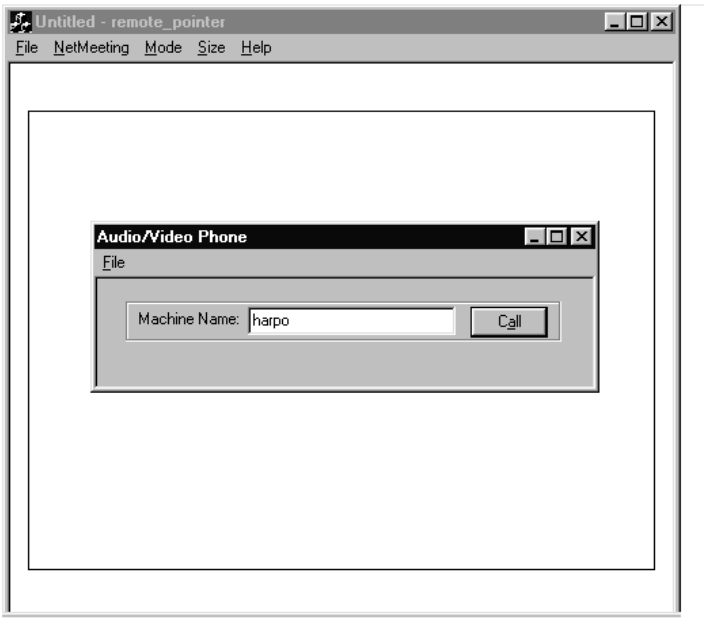


FIGURE 14. Call Dialog

To invoke remote pointing, both participants have to choose the ‘Remote Pointing’ option from the ‘NetMeeting’ menu (Figure 13). In a production system that could be done automatically.

A NetMeeting data channel is set up that transmits the position of the mouse cursor from the desktop computer to the wearable computer.

An additional red cursor is overlaid over the video image (Figure 15) on the HMD to represent the position of the mouse cursor controlled by the desktop user. When the desktop user presses the mouse button, the video image is frozen and the cursor moves over the still video image allowing exact pointing. A small extra window not shown here continues to provide the live video stream.

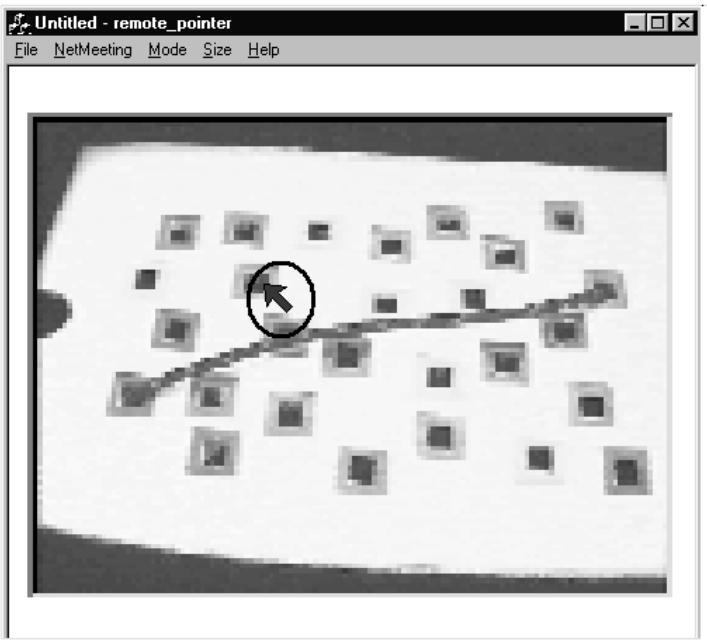


FIGURE 15. Video Image and Remote Pointer

Problems Encountered

The main problem we had with NetMeeting was that we couldn't get direct access to the video image. As the image is being updated all the time, we can't just copy our remote pointer on top of it. All we would get is some flickering. Therefore we decided to get a handle to the picture with the video image. From there, we would periodically copy it into our window and copy the remote pointer on top of it. This has the additional advantage that we can change the size of the video image as we like, albeit the quality is deteriorating, if we make it too large.

As we have the original image in an extra window, we can watch the live video there while the main image is frozen.

There was another problem which took us a lot of time to solve. When changing the capabilities of a call from video/audio only to video, audio and data, NetMeeting wouldn't create the channels anymore. We finally managed to get the channels again by placing a sleep command after creating the call. So, NetMeeting needs additional time to set things up when adding data capabilities and it doesn't wait automatically.

The latest problem we encountered was that different versions of NetMeeting 2.1 show different clippings of the video image. This means, our remote pointer may be off and we need some method of calibration. Since the problem does not occur when using the same version of NetMeeting, it wasn't relevant for our prototype system.

CHAPTER V

USABILITY STUDY

After designing and implementing a prototype system, we wanted to evaluate it. Therefore we conducted a usability study that is presented in this chapter. A full-fledged usability study testing the NETMAN system within its intended scenario is beyond the scope of this thesis. Instead we performed a limited form of test that focussed on the evaluation of remote pointing.

We believe that remote pointing works in all sorts of scenarios. So far, we have given three examples, the hiker and the ranger, the architect and the civil engineer, and the field technician and the expert in the NETMAN scenario. As exemplified in the hiker and ranger scenario, we see remote pointing as the natural equivalent of pointing when we have a computer-mediated communication of the form we illustrated in previous chapters.

Pointing is an important element of communication in real-world scenarios. It is used to focus on a point of interest in an environment that is so complex that it would require a lot of words to express the same information verbally. We expect that remote pointing will be able to serve a similar purpose as pointing.

Hypotheses

In order to clarify our expectations about remote pointing and the prototype system, we formulated the following hypotheses:

1. Users will prefer remote pointing to voice-only communication. This means that most often the decisive element for describing the next step in the collaboration will be remote pointing.
2. The combination of remote pointing and verbal communication is faster than just verbal communication. Remote pointing allows us to quickly focus on the point of interest.
3. Desktop users will use freezing quite often to counterbalance the wearable user's head movement.
4. Users will use the secondary window to observe the wearable user and make sure the wearable user is doing things right even though the main window remains frozen to allow effective pointing.
5. The number of errors should be minimal if our design really mimics real-world pointing. (We call it an error, if the communication breaks downs or a task is not performed correctly.)
6. Language will be used in the same way as in real-world situations: In 'difficult' situations it is used to augment pointing. In less difficult situations it will be used less.

Our goal for the usability study was to find out, if our hypotheses about remote pointing are valid. The task we came up with is derived from the NETMAN scenario. A common problem in network maintenance is that cables are not plugged into the correct outlets. Imagine a network technician standing in front of a wall of routers and switches with hundreds of cables and outlets. Given that he is connected to an expert, the expert can tell him/her where a new cable has to be plugged into. This is a typical example for remote pointing, because a description with words alone is possible, but may be far more difficult. Based on this situation, we developed an artificial task that allowed us to vary the level of difficulty.

Task

In the actual study, we had two people collaborating in building a *mockup* circuit using our prototype system. We had a desktop user who played the part of the expert and a wearable user who played the part of the network technician. The desktop user had a *wiring plan*; he/she had to guide the wearable user through setting up the circuit.

For the first part of the test, the participants had to rely on video and audio only for their conversation. In the second part, they were instructed to use as much remote pointing as possible. This allowed us to compare their communication with and without remote pointing. In the following parts, it was up to them how to communicate in the most efficient way.

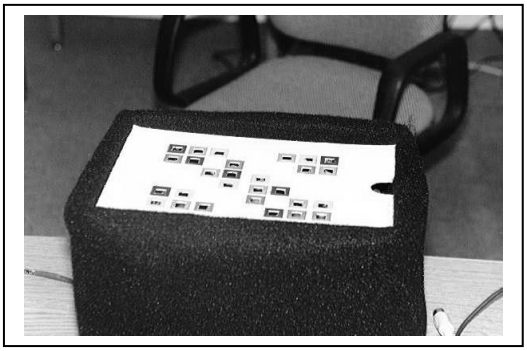
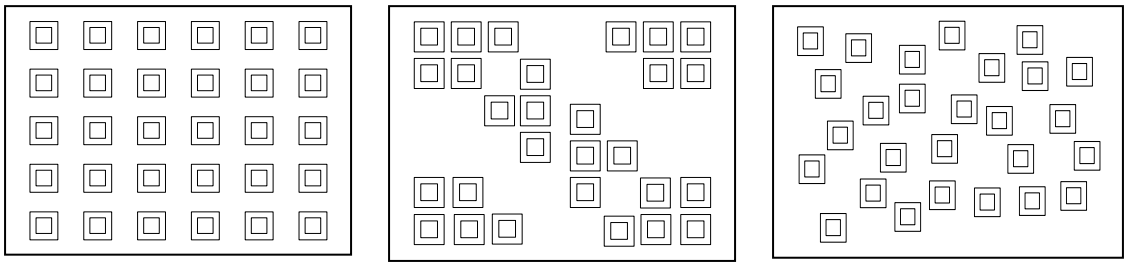


FIGURE 16. Cardboard Box

The mockup circuit had to be plugged together on top of a cardboard box. (Figure 16) We could replace the top part of the box with different cardboard templates. The templates had square holes cut into them that were arranged in different ways. Instead of real cables, we used colored cords. We still stick to our ‘circuit’ terminology here and call them ‘cables’.

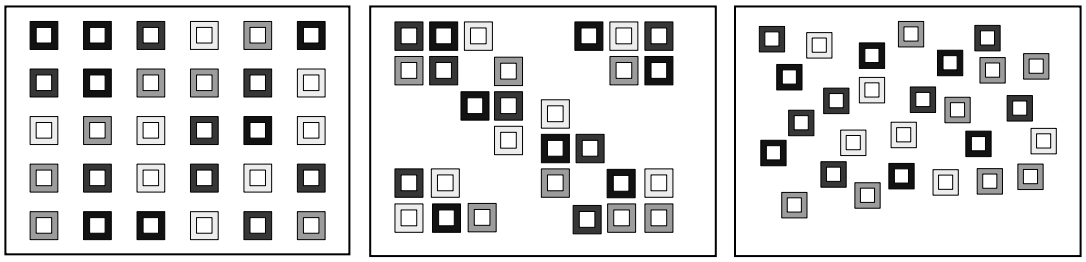


Grid Structure Chaos

FIGURE 17. White Templates

In designing the templates, we created a number of difficulty levels. (Figure 17 & 18) The levels were more relevant for verbal descriptions than for remote pointing. Our

templates differed in the coloring and the arrangement of the holes. The colored templates had holes coded with yellow, red, blue and green, while the other templates were just white. The holes were ordered in a grid arrangement, a structured arrangement or a chaotic arrangement. Taken together we used six different templates in the study.



Grid Structure Chaos

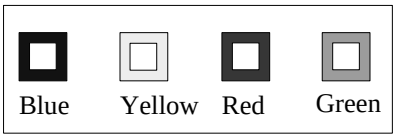


FIGURE 18. Color Templates

All the cables had the same length. We used four different colors: white, yellow, blue/red, and blue/green.

Distinguishing the blue/red and the blue/green cable on the video image was difficult. It was also hard to see the white cable on the white cardboard, especially if holes were not color-coded. These difficulties were sort of intentional; they might exist in real-world tasks, so we wanted to see how our participants dealt with it.

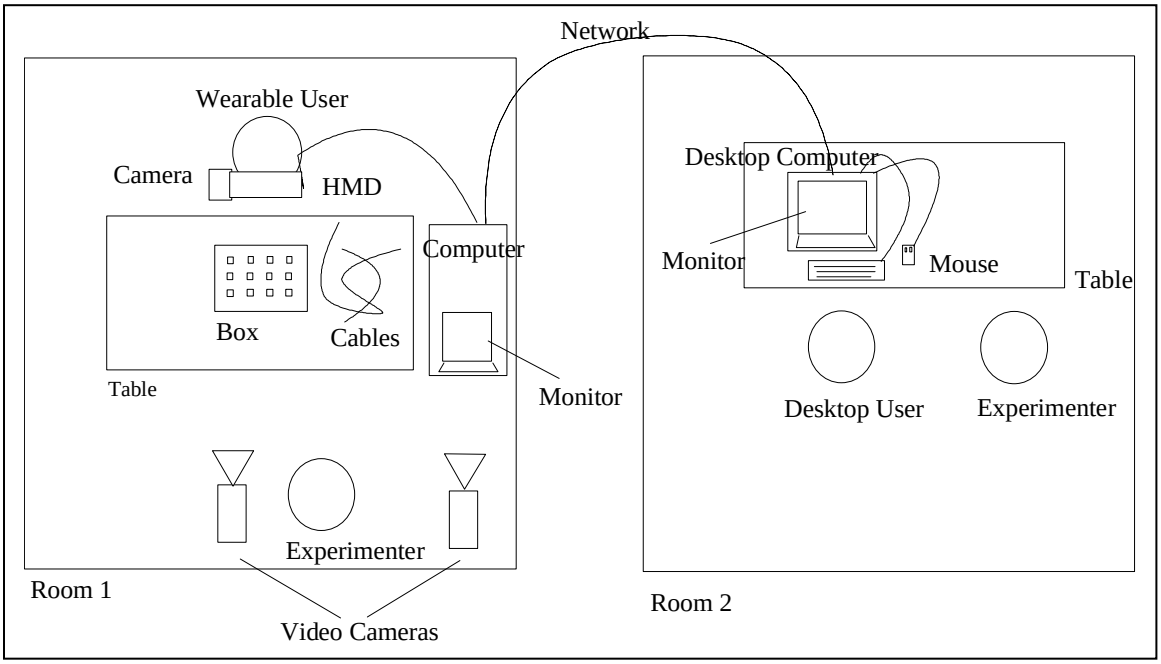


FIGURE 19. Study Setup

Setup

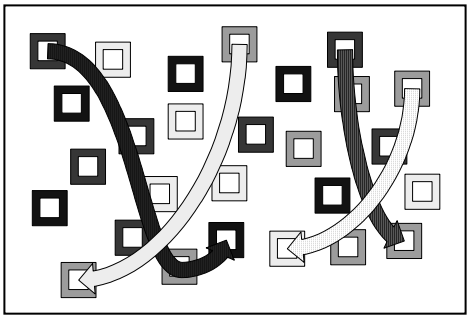
The desktop user and the wearable user were working in separate rooms. There was no direct communication; all communication was mediated through our prototype system. Figure 19 shows a plan of the setup in the two rooms as it would be seen from above.

Wearable User

The wearable user was standing or sitting in front of a table, whatever he/she felt was more comfortable (Figure 19, Room 1). He/she was wearing a head-mounted display (HMD). We used the i-glasses! that were described in Chapter IV. We operated them in see-through mode with the left eyepiece removed. Attached to the head-mounted

display was a microphone for voice communication and a video camera pointing in the direction in which the user was looking. The image of the video camera could be seen on the HMD, as our software was running. The video image gave a slightly different view from what the user saw directly through his eyes. The reason was that the camera was attached to the side of the HMD, so the center of the eye and the center of the camera were about three inches apart, so the perspective was different. In addition, the camera had a wider focus.

For practical reasons we used a tethered version of the system. This eliminated the need to change batteries during testing. Therefore, the HMD was connected to a desktop computer. A VGA monitor was also connected to the computer. This allowed us to videotape what the wearable user was seeing on the HMD for later evaluation.
The box, on which the circuit was built, was placed on the table. The cables could be found next to it. During the test the wearable user had to plug the cables into the holes guided by the desktop user.



Desktop User

The desktop user was sitting in front of a desktop computer in the other room (Figure 19, Room 2). He/she had a headset with a microphone, so he/she could talk to the wearable user using the audio connection. The desktop computer was running our software, so the desktop user could see the video image of wearable camera on his screen. As it is commonly the case, a mouse was connected to the computer. Moving the mouse cursor on the video image moved the red remote cursor on the wearable user's display. The cursor was positioned on the same coordinates relative to the video image as the mouse cursor on the screen of the desktop computer.

The desktop was given a wiring plan for each subtask. Figure 20 shows an example. During the test he/she had to instruct the wearable user how to set up the circuit according to the wiring plan.

Participants

Nine pairs, or 18 people altogether, took part in our study. We recruited them through personal contacts and posters that were placed in various locations throughout the computer science department. The participants could choose their own partner for the study, if they wanted to. Otherwise, we paired them with another participant.

Seven participants were female, eleven were male, and we had one mixed pair. All participants stated that they had normal color vision and no other uncorrected vision problems. In an entry questionnaire, we asked them about their computer background and prior experience in fields related to this study.

About three-fourth of the participants were students of computer science or related areas like mathematics or electrical engineering at the University of Oregon. Two participants were studying other subjects and two were non-students.

All of them rated their computing ability and experience between three and five on a scale of five with five being the expert level. Most of them had a vague idea what wearable computing was, with most of the information coming from magazines and other news media. Four participants had used a video conferencing application before and six had experience with collaborative software ranging from computer games to whiteboard applications. Only one participant had experience with a head-mounted display, which had been used for virtual reality games.

So, in general, our participants had good basic skills in using computers, but little experience with applications related to the field of our study.

All participants signed a consent form informing them that participation was voluntary, that they could terminate the study at any time and that all personal information would be kept confidential.

At the end of the session we offered cookies, and, as an incentive, we raffled a dinner for two.

Procedure

Each test consisted of the following steps:

Both participants were given a handout containing some information about the goals of the usability study and a short description of the tasks that they were going to

perform. After reading the handout, the participants were asked to read and sign a consent form and to fill out the entry questionnaire mentioned above. All the questionnaires can be found in Appendix B.

Then both participants were shown around both work areas and they had the opportunity to ask questions about the test. We tried to make sure that they had understood how the test was going to be conducted.

We left it to each pair to decide who wanted to be the wearable user first. We switched roles about three-forth through the test, so everybody got the chance to see both sides.

We had an *experimenter/investigator* in each room. It was their job to help the participants with the equipment, e.g. put on the HMD, prepare the different subtask, e.g. make sure that the desktop user has the right wiring plan, and fill out a protocol from that provided us with some information for the evaluation of the study. (Samples of the protocol form can be found in Appendix C.)

In addition to the protocols and questionnaires, we videotaped the test for later evaluation. We had two cameras. One was directed at a monitor showing the same image as the head-mounted display and the other, recorded as picture in picture, was pointing at the wearable user and what he was doing.

To start with, we gave the participants a warm-up task that enabled them to familiarize themselves with the equipment. The wearable user looked around in the room and the desktop user picked an object (e.g. the telephone). He froze the picture and

pointed at the object. The wearable user had to identify the object. The users were asked to repeat that a couple of times until they felt comfortable with using remote pointing.

For the first section of the test, the participants had only audio and video to accomplish the two subtasks. This means they had to communicate verbally to set up the circuit. The first subtask was based on template with a white grid, the second on one with a structured arrangement of the color-coded holes. In the second section of the test, the participants were asked to use remote pointing whenever possible. The same templates were used as in the first section, but we had different wiring plans. After the first two sections, the participants were asked to fill out 'Part I' of the questionnaire that asked them to compare the tasks under the two different conditions.

The four subtasks of the third section were based on the four templates that had not been used so far. The participants were asked to accomplish the tasks efficiently with whatever combination of remote pointing and voice communication they considered appropriate.

After the third section the participants changed roles. The warm-up task was repeated to give them an opportunity to get used to the equipment they had to use for the new role.

The fourth section repeated the third section with different wiring plans.

The following 'Part II' of the questionnaire asked about how useful remote pointing was for the different subtasks and what the participants thought the share of remote pointing and verbal communication was for the last two sections. The exit interview asked the participants for their impression regarding the usefulness of remote pointing in

general and our implementation in particular. They were encouraged to make suggestions how the implementation could be improved.

Results

We used three different sources for our evaluation - the questionnaire, the protocol sheet that the experimenter filled out during the test, and the video recordings.

The questionnaire provided us with the subjective opinion of our participants. This is very valuable feedback, as the system will only be used in practice if the people it is intended for see an advantage in using it.

The purpose of the protocol sheet was mainly to record the impressions of the experimenters during the sessions. They gave us a good starting point for our further investigations. The evaluation sheet we used can be found in Appendix D.

Watching the video recordings, we tried to capture some objective data about how people used our system. Here we focused on different aspects of communication, especially the way remote pointing was utilized and how important it was for the conversation as a whole.

Discussion of the Hypotheses

At the beginning of this chapter, we presented some hypotheses we wanted to prove. In the following we discuss how successful our usability study was in this regard.

1. *Users will prefer remote pointing to voice-only communication.*

   In order to investigate this hypothesis, we look at data pertaining to the usage of remote pointing that we collected by evaluating the video recordings. We also look at the results of the questionnaire to find out about the users' opinions.

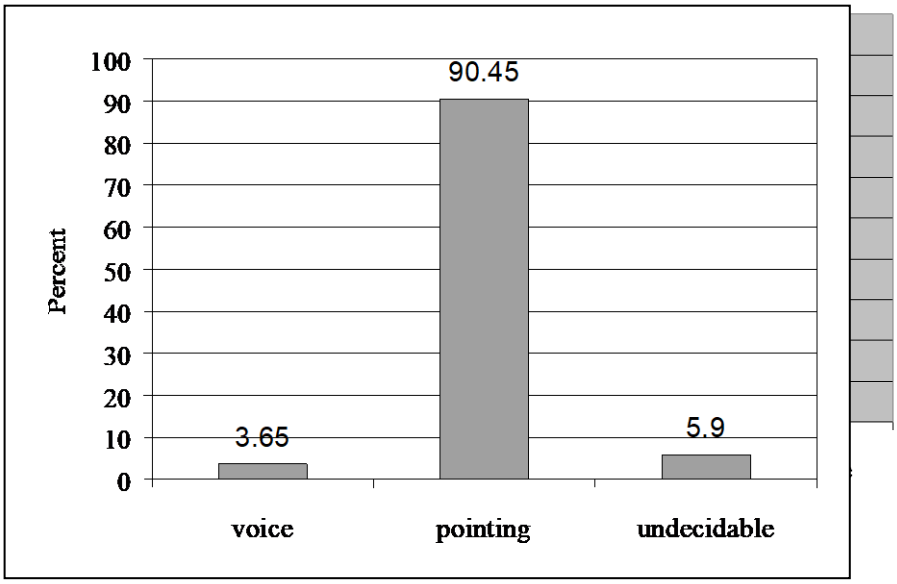
   As we watched the video we counted the number of times, in which voice or remote pointing was the decisive element for finding the hole to plug the cable in. We define the *decisive element* as the communicative element that first establishes the identity of the object being pointed at, so that the remote user can act upon this information. When watching the video, we determined the moment, when the wearable user started moving towards the target, or more precisely, when we saw a directed movement in the small video window. We then identified the relevant information that was available at that moment to find the decisive element.

   For subtasks five to twelve, it was up to the users to decide how to communicate most efficiently. We found out that remote pointing was used in 99.3% of the cases. In 521 (90.45%) cases, remote pointing could be determined as the decisive element, in 21(3.65%) cases it was verbal communication and in 34 (5.9%) cases it couldn't be decided. (See Graph 1; for more numbers see Table 4 in Appendix E.)

   This gives us a strong indication that users readily accept remote pointing as a tool that helps them to accomplish a given task more efficiently.

   When asked directly 100% of the participants answered the question, if they had enjoyed using the system with yes, 100% thought it could be useful in other scenarios.

Although there were some problems, which we discuss below, these numbers nevertheless point to a high “user satisfaction”. This provides strong evidence in favor of our hypothesis.



GRAPH 1. Decisive Communication Element in Subtasks 5-12

2. *The combination of remote pointing and verbal communication is faster than just verbal communication.*

   Looking at the way people solved the tasks, we got the impression that the use of remote pointing speeds up communication considerably. As we expected, it seems to provide a quick way to focus on the point of interest. This is also supported by the fact that remote pointing was used so extensively as shown in the previous paragraph. People wouldn’t do that, if there was nothing to gain. For our simple tasks, time would be the most important gain that was achievable.

The way we conducted this study, it is not possible to give any numbers concerning a "speedup" that was achieved. There are so many factors involved in human communication that the number of people who participated in the study would have been too small to make any meaningful statement.

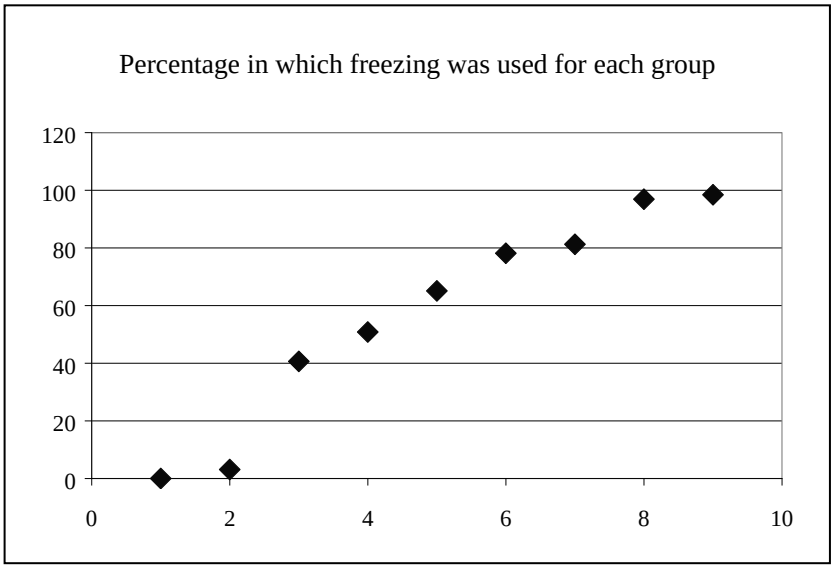
It is interesting to note that the pairs that relied almost exclusively on remote pointing as their means of communication appeared to be the fastest.

3. *Desktop users will use freezing quite often to counterbalance the wearable user's head movement.*

   Freezing was used in 56.8% of the cases. The usage of freezing differed considerably between the groups. As shown in Graph 2, there were two pairs that practically didn't use it at all (0%-3%), two groups used it all the time (96%-98%) and the other five groups fell somewhere in between (41%-81%). The exact numbers can be found in Appendix E, Table 5.

   Part of this huge variance may be due to problems with the implementation of freezing. Occasionally desktop and wearable user wouldn't get exactly the same frozen image. Therefore the pointer would point at the wrong position on the wearable user's image. A more detailed discussion of the problem can be found below.

   Some users who experienced these problems may have decided to stop using freezing entirely. This is supported by the fact that one group used freezing all through subtasks three and four, but didn't use it for the rest of the subtasks, which



GRAPH 2. Distribution of How Much Freezing Was Used in Each Group

form the basis of this discussion. That is why it is difficult to make any general statement about freezing at this point.

Some users who didn't utilize freezing implemented it in their own way. The wearable user kept the head still, so the camera image would remain stable to allow pointing.

Even those who used freezing didn't all use it in the same way. Some people froze the image only for a very short time (less than two seconds), others for longer periods. There were even a few pairs that developed the 'permanent freeze' method. They froze the image once at the beginning of the subtask, when a new template had been put on top of the box. They kept it frozen until the last cable was plugged in and released it only for a final check.

4. *Users will use the secondary window to observe the wearable user and make sure the wearable user is doing things right even though the main window remains frozen to allow effective pointing.*

   This behavior is difficult to quantify. However, sometimes it is noticeable in the conversation, e.g. when the wearable user asks if he just plugged the cable into the right hole and the desktop user confirmed even though the main image was still frozen. It was especially obvious for the pairs that used the 'permanent freeze' method described above.

5. *The number of errors should be minimal, if our design really mimics real-world pointing*. (We call it an error, if the communication breaks down or a task is not performed correctly.)

   There were two problems that occasionally led to errors. The most important one was the freezing problem that we mentioned above. We counted twenty out of about 700 pointing actions, in which the pointer was noticeably off. There may have been other cases, in which the pointer was slightly off, but still pointed at roughly the right location. The second problem has not much to do with remote pointing. It has to do with the colors of the cables. The blue/red and blue/green cables were hard to distinguish on the video, so they were sometimes confused. We counted twelve such cases. Except for one group, the participants noticed the problem and corrected the error. The white cable is very hard to see on the white templates, especially if the holes are not color-coded. Sometimes that lead to confusion.

In general, errors were detected and corrected. Therefore the tasks were completed satisfactorily in the end. If we disregard the problem with image freezing, there were hardly any communication problems when remote pointing was used. So we believe that after fixing the problem this hypothesis should be satisfied.

6. *Language will be used in the same way as in real-world situations: In 'difficult' situations it is used to augment pointing. In less difficult situations it will be used less.*

   In general, the conversation was very one-sided. The desktop user gave the instructions; the wearable user was following them. In the cases in which remote pointing was used, instructions often consisted only of what we would like to call *referential statements*, e.g. "Put the cable in here", "The green box there", etc. In obvious cases, an absolute description was given, e.g. "the hole in the upper left corner". The rest of the conversation was restricted to a few *confirm* questions, e.g. "The blue hole?" and some acknowledgements. Some people accomplished the tasks without any additional verbal instruction.

   The conversation pattern changed only if there was a problem. Then the number of utterances increased and the share of the communication was more evenly divided among the participants.

   Some pairs agreed on a coordinate system before starting out, especially for the first two tasks that had to be solved with audio and video only.

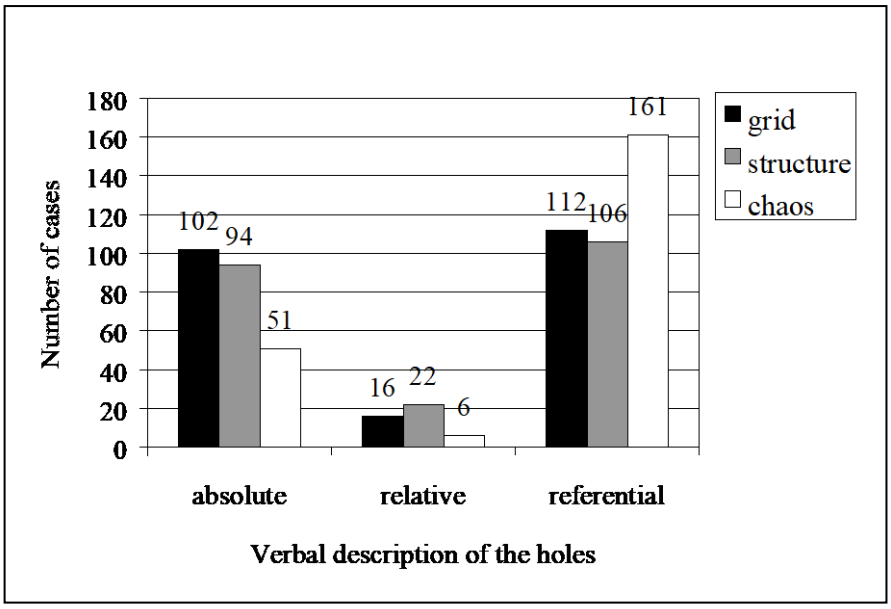
After discussing our hypotheses, we now look at other results of our study. We start with the effects of the difficulty levels on remote pointing and communication. Then we investigate the causes of the freezing problem. This is followed by a discussion of the user feedback we got regarding other problems and the users' suggestions for further improvements.

Difficulty Levels

It is interesting to see that the different templates (white or color - grid, structured or chaotic arrangement) that were intended as difficulty levels didn't seem to have an influence on the way remote pointing was used. The number of pointing actions had the same high level in all the different situations. We counted between 284 and 286 pointing actions out of a maximum of 288 for the respective tasks.

As we had expected, the communication differed between the arrangements. We classified the utterances that were used to describe the location of a hole into three categories. If it was completely described, e.g. "The second hole to the left of the lower right corner", we called that an *absolute* description. If the location was described relative to a hole, in which a cable had been plugged previously, we called that *relative*. Finally, if the description was only used to support the pointing, e.g. "plug it in here" or "there", we called that *referential*.

As can be seen in Graph 3, the number of absolute descriptions is much higher for the grid and structured arrangements than for the chaotic arrangements. (See Table 6 in Appendix E for more results.) It is much easier to completely describe the location of a



GRAPH 3. Types of Utterances Used for Different Arrangements of the Holes on the Templates.

hole that is part of some structure than a location in some arbitrary arrangement. In the latter case, the participants used more referential descriptions.

In order to get comparable numbers, we only used data from subtasks five to twelve. As there were twice as many tasks using chaotically arranged templates, we multiplied the numbers for structured and grid arrangements by two.

The Problem with Image Freezing

During our usability study we discovered a problem with image freezing. Occasionally the remote pointer seems to be off from the point of view of the wearable user. It is not pointing at the correct location. We soon found the reason behind this problem: When the desktop user presses the mouse button, his/her screen is frozen and a freezing command is sent over the network to the wearable computer. The wearable computer freezes the image on arrival of this message. In case of a longer than usual delay, e.g. a high network load, the wearable computer may already display the next video frame. If the wearable user has just turned his/her head a little, the desktop user and the wearable user see different images and the pointer points at the wrong location.

The problem remained unnoticed partly because we developed the program on a desktop computer. There, we had a fixed camera, so the only movement could come from the user. But in this case, the user was mostly looking at the screen and not moving very much. This is why the image didn't change much and the pointer couldn't be off.

The discovery of this problem was a useful outcome of our study. In a future version of the system, we will probably send back the image on which the desktop computer has frozen to guarantee that both users see an identical image.

- It is difficult to focus on the grid with one eye closed.
- It is hard to see anything on the head-mounted display, unless you have a solid background.
- The head-mounted display is heavy.
- The perspectives of the camera and the eye are a bit off, but once you get used to it, it becomes a lot easier.
- The video needs to have a higher resolution; the audio was good, though.
- It was difficult to switch between eye viewer and the task at hand. The head assembly obstructed the eye used for working.
- Sometimes there were interruptions in the audio. The fingers got tired from pressing the mouse button when freezing the picture.
- It was difficult to verify the actual placement of the cables, probably because of a lack of resolution.
- Sometime it was difficult to hear what my partner was saying.
- The resolution made it difficult to check the results.
- If the pointer was less blocky, it might be easier to distinguish exactly what is being pointed at.
- The red color makes [the pointer] stand out, but it is quite large, so it is not as specific for pointing at a refined area.
- It would be easier to point, if the frame rate was better.
- It might be better to freeze the screen by pressing a key on the keyboard rather than the mouse. Some people might associate clicking the mouse button with other actions and forget to unfreeze the screen.
- Aside from the refresh rate, it was very easy to point at targets.
- The large pixel at the pointer tip made it difficult to tell exactly what was being pointed at.
- I had some focussing problems with my eye.
- My eyes and neck were tiring after a while.
- It was a little bit hard to see what the monitor was showing and to match it with the actual grids.
- The only problem was the transmission delay.
- A higher resolution would be nice.
- A helmet-style remote device would be good in terms of comfort.
- It would be good to have a better resolution for the camera to deal with small objects.

FIGURE 21. User Comments

In Figure 21 we present a list of the problems identified by our study participants. We summarize them into six different aspects and discuss them in the following. We

excluded comments regarding the image-freezing problem or the colors of the cables that have already been discussed above.

1. The perspective of the camera and what the user sees through his/her eyes is not identical. Some users found it difficult to focus on both the real world and the camera image as displayed on the head-mounted display.
2. Users felt that the head-mounted display was heavy. One user complained that the neck was hurting after a while.
3. Two people indicated that sometimes it was difficult to understand what the co-user was saying.
4. Three participants thought that the remote cursor was too "blocky", so it was difficult to tell exactly what was being pointed at.
5. One user said that his/her fingers got tired from pressing the mouse button. (The user was part of a group that used permanent freezing.)
6. A lot of people thought that a higher resolution of the video image would be helpful, some wanted a higher frame rate and others again thought that there was too much delay.

The first three points are related to the equipment we used. It was clear from the start that our equipment couldn't compete with the top products on the market. Due to the high cost of new equipment, we were constrained to use the equipment on hand. Our head-mounted display and our camera are rather bulky. It wasn't possible to attach the camera at any better location than the side of the head-mounted display (see Figure 8). This meant that the perspective of the eye and the perspective of the camera would be

significantly different. If we had a small camera, we could mount it somewhere between the eyes, reducing this problem. Further, we'd like to experiment with a small head-mounted display with a better resolution that is mounted slightly above the eye. This would give the user an unobstructed view onto the workspace, but would allow him to look up to the display when necessary. We believe that this equipment would solve some of the problems mentioned by the participants, at least it would ameliorate the situation.

We don't think that the audio connection itself is a problem. The microphone we attached to our head-mounted display has to be really close to the mouth to produce a reasonable signal. People also need to speak up a little. It is possible that the conditions were not ideal for those two users, although we always tried to make sure that everything was working properly.

When we designed the remote cursor, we wanted to make sure that it was very easy to spot. Therefore we made it slightly larger than a normal mouse cursor and colored it red. Three users thought that this was too blocky, but 94.4% of the participants answered the question, if form and color of the remote pointer were appropriate with yes (including some of those who remarked that it was too big!). We need to investigate the issue of pointer form and size further to make any final statement about it.

We hadn't considered using 'permanent freezing', a method developed by two groups, in which the picture was frozen at the beginning of the task. It would not be unfrozen until the whole task was completed. It is not surprising that the fingers of desktop users get tired if they have to press the mouse button for such an extended

period. If we want to support permanent freezing, we have to find another method, e.g. clicking a button for freezing the image and another for releasing it again.

Most people would have liked to have better video quality, be it the resolution, the frame rate or the delay. The problem we face, which is typical for wearable computing in general, is that the bandwidth of our current wireless modem is restricted. This means we have to find the right balance between resolution and frame rate, so that we achieve a usable result over our wireless connection. As the results of our study show, it is possible to work with this somewhat limited resolution and frame rate, although better conditions would always be desirable.

Frame rate and delay varied during our study, depending on the network congestion in our computer science department. This is somewhat unfortunate, because it affected some participants more than others. On the other hand, we would have to cope with the same effect in real world situations.

Suggestions

There were three suggestions to extend the given functionality:

1. One user suggested providing an extra cue, e.g. a sound, when freezing the image.
2. Somebody else suggested drawing a line after clicking the mouse button that could be pulled towards the destination.
3. The third suggestion was to give the wearable user the option of freezing we image.

The first suggestion seems to be helpful to catch the attention of the wearable user. It provides a signal that the desktop user is going to point at something. We therefore plan to include it in a future system.

If drawing a line improves the communication needs to be investigated further before we come to a conclusion.

As far as the third suggestion is concerned, the user already has the option of freezing the video image. He just needs to click a button under the small video window to stop sending. This functionality is provided automatically by NetMeeting and therefore hasn't been discussed so far. Pressing that button will stop the live video, but leave the last image on the screen. So, once the wearable user has an input device, he/she can freeze the image this way.

Summary of the Results

Our study showed that, although there were some technical difficulties and a problem with our implementation of image freezing, remote pointing was overwhelmingly used by our participants. In most cases, remote pointing was the decisive communication element for determining a location. As we have seen, it provides a fast way for a desktop user to focus the attention of the wearable user on an object in his/her physical space.

The results of the questionnaire show that the participants thought the system was useful and enjoyed using it. At this point, we want to add two numbers to the ones already given that support this claim. The participants gave our remote pointing

implementation an average of 4.52 on a scale from 1=useless to 5=useful with a standard derivation of 0.61. They rated the difficulty of both pointing in the role of the desktop user and figuring out what was being pointed at in the role of the wearable user as approximately 1.8 on a scale from 1=easy to 5=difficult. The standard derivations were 1.0 and 0.7 respectively.

CHAPTER VI

CONCLUSION

We presented remote pointing as a way of porting the natural gesture of pointing into the world of collaborative wearable computing. The main focus of this thesis was to find out how effective and efficient remote pointing is.

In order to do this, we designed and implemented a prototype system. Since it is impossible to point at objects when the video image is changing constantly due to the movements of the wearable user, we gave the remote user the option of freezing the image. Then he/she could move the pointer on a still image, allowing exact pointing.

We used the prototype system to conduct a usability study with real users. We designed a simple artificial task that enabled us to evaluate the remote pointing aspect of the collaboration.

We found out that remote pointing indeed offers a quick way of focussing on an object. Our users liked the system and basically used remote pointing whenever possible, so it played a dominant part in their communication.

Although the idea of remote pointing in a wearable setting may not be new, we were the first to conduct a usability study showing the effectiveness and efficiency of remote

pointing. Using image freezing to allow pointing even though the view is changing appears to be a natural solution, but hasn't been used in this context.

A shortcoming of our current implementation of image freezing is that occasionally the users will have different frozen images. This makes exact pointing impossible. In a future version this problem will be eliminated.

The next step in the evaluation of remote pointing would be to test it on a real-world task, perhaps as part of an evaluation of the fully implemented NETMAN system that we have discussed in Chapter III.

As shown in Chapter II, remote pointing can be seen as one of several collaboration primitives. We are currently working on a system that uses another collaboration primitive, remote sensing, to augment a remote user's perception of another user's physical environment with sensor information. Such sensors could provide information about location, identity, medical conditions or the state of machines.

We believe that wearable computing provides a platform for a whole set of new applications that allow collaboration in a physical environment. Remote pointing is one step in that direction.

APPENDIX A

PROGRAM STRUCTURE

This appendix just gives a short overview over the files that constitute the actual implementation. We give the class/classes that is/are associated with each file and a short description of their function. The files can be divided into three different classes.

The first class consists of the files that are derived from MFC classes and were created by the Class Wizard of Visual C++ (Table 1, see next page).

The second class consists of the files that pertain to the implementation of our NetMeeting functionality (Table 2). They are listed only for reason of completeness and to give a mapping between files and corresponding classes. The objects involved have been discussed in depth in Chapter IV.

The third class is made up of utility functions that didn't fit in any of the other classes (Table 3).

TABLE 1. Files Created by the Visual C++ Class Wizard

Standard MFC Files created by the Class Wizard	Associated Classes	Description
MainFrm.cpp/h	CMainFrame	CMainFrame is derived from the MFC class CFrameWnd. It implements the frame window. In this case, it is the main 'single document interface' (SDI) window.
Remote_pointer.cpp/h	CRemotePointerAPP	CRemotePointerAPP is derived from the MFC class CWinApp. It implements the core application, the message dispatch and the command-line processing. It creates and destroys Documents and Views.
Remote_pointerDoc.cpp/h	CRemotePointerDoc	CRemotePointerDoc is derived from CDocument. It contains the document implementation that manages information handled by application.
Remote_pointerView.cpp/h	CRemotePointerView	CRemotePointerView is derived from CView. It contains the view implementation that provides the user with a visual representation of the information.

TABLE 2. Files Related to NetMeeeting

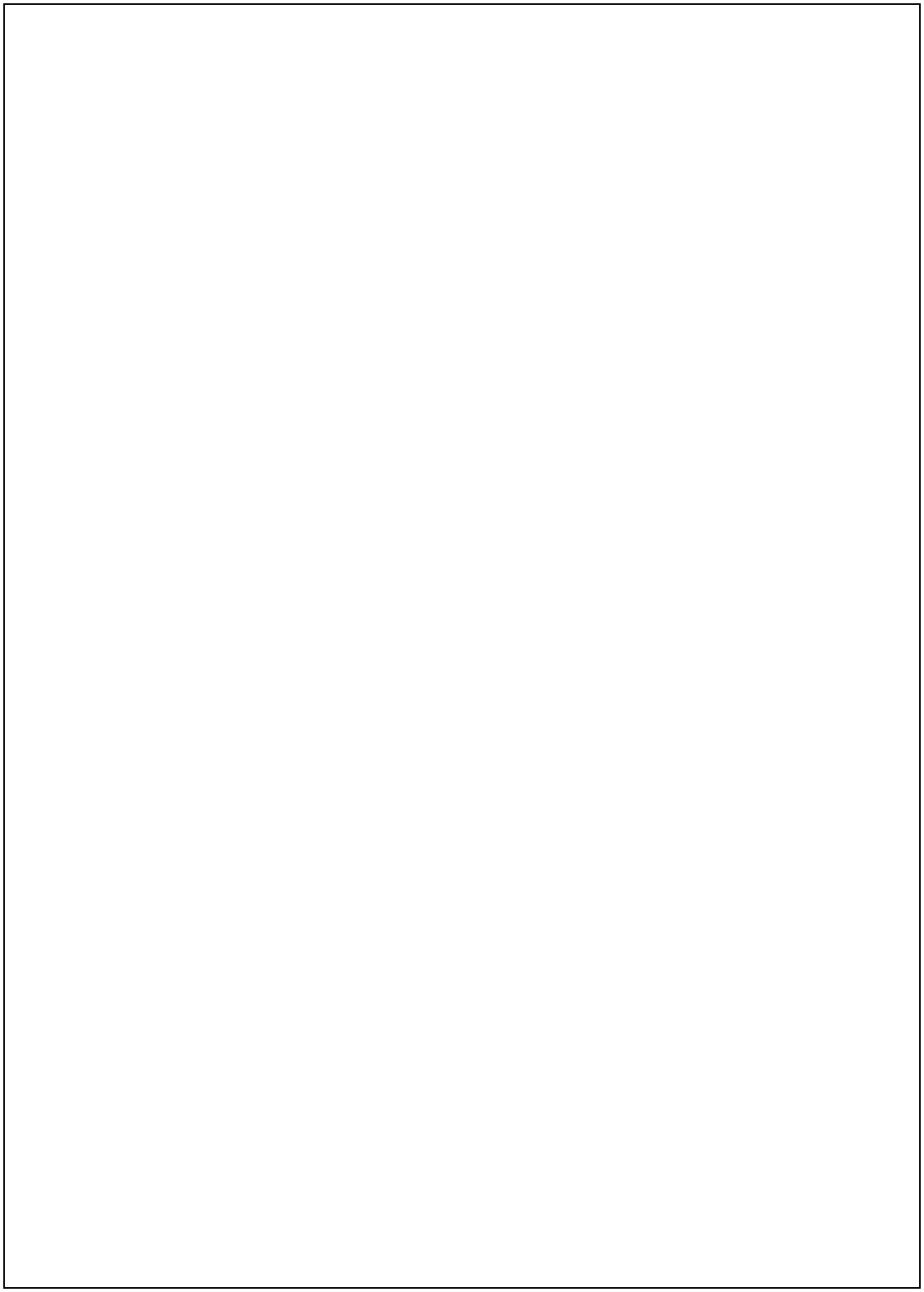
NetMeeting Related Files	Associated Classes	Description
Callntfy.cpp/h	CCallNotify	Notification sink for calls.
Cdata.cpp/h	CDataNotify	Notification sink for our data channel.
Clutil.cpp/h	RefCount/CNotify	Classes that provide the basic functionality required for COM Object. All the other classes inherit the methods.
Cnfntfy.cpp/h	CConfNotify	Notification sink for conferences.
Conf.cpp/h	Conf	Wrapper class for everything that has to do with conferences, calls and managing them.
Mgrntfy.cpp/h	CMgrNotify	Notification sink for the conference manager.

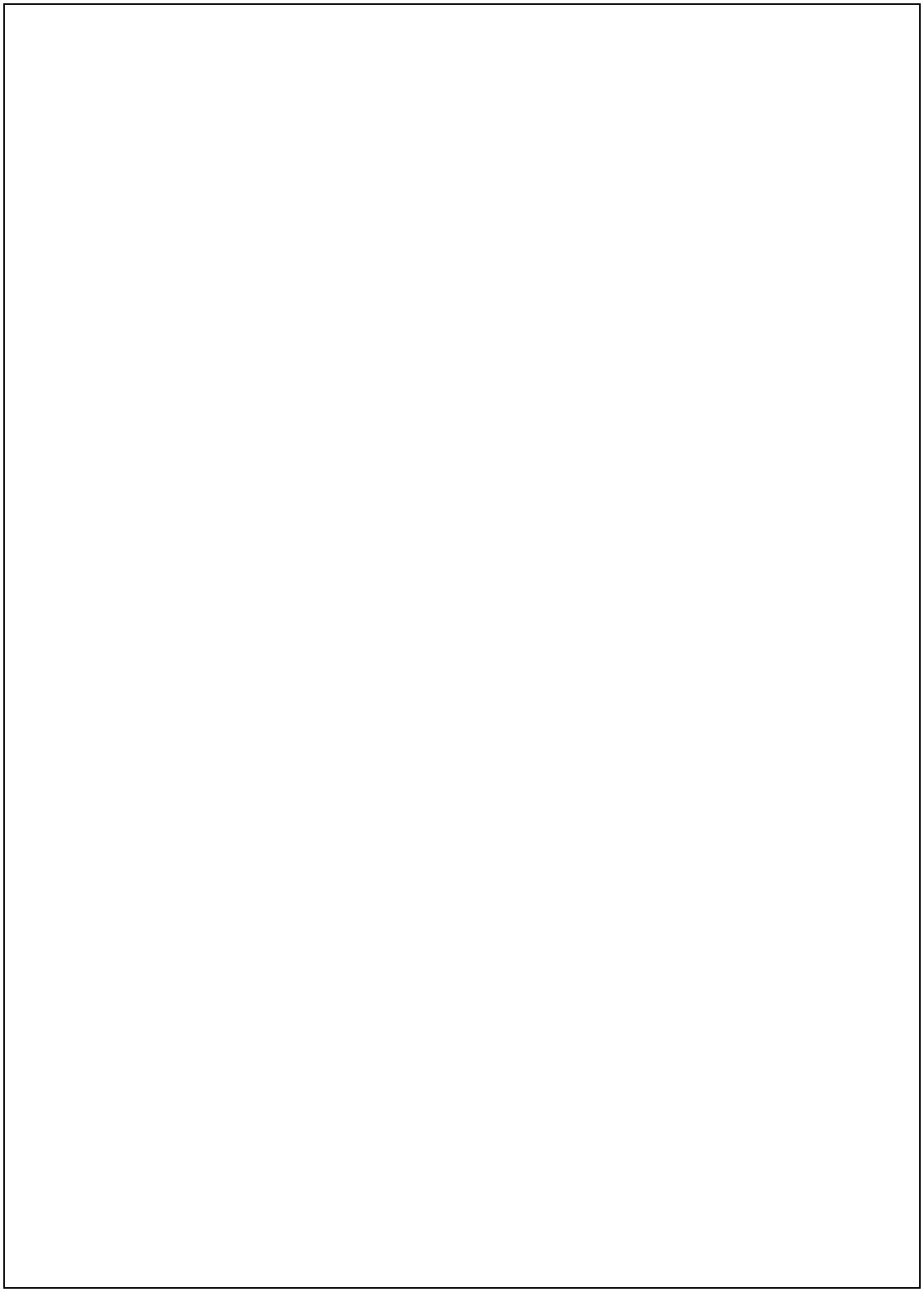
TABLE 3. Other Files

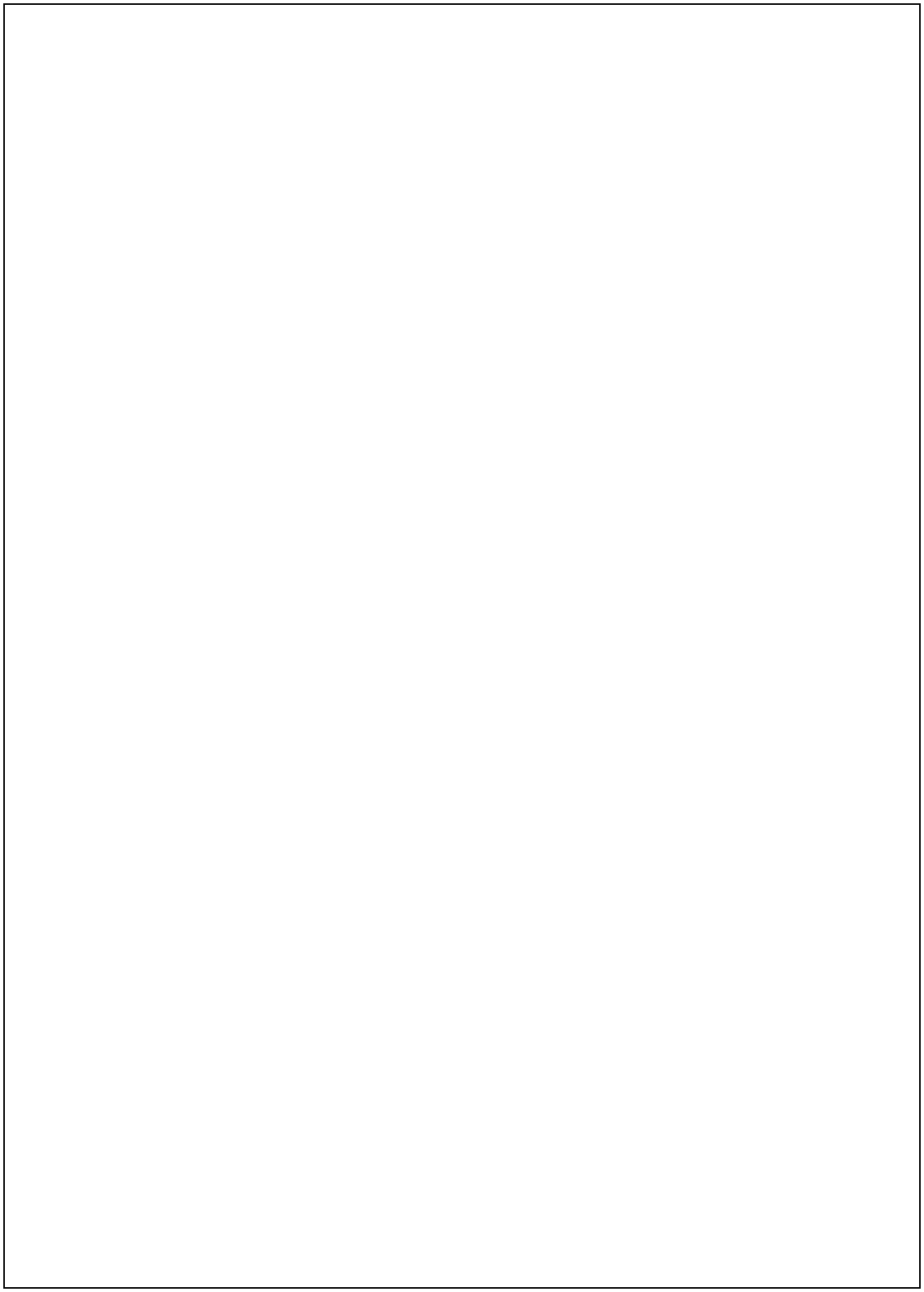
Utility Files	Description
FindWindowUtil.cpp/h	Provides functions to get a handle for windows and their children given the name of the window. We need them to get a handle on the video window, because NetMeeting doesn't give us direct access to the video stream.
Util.cpp	Routines for sending and receiving coordinates of the remote pointer.

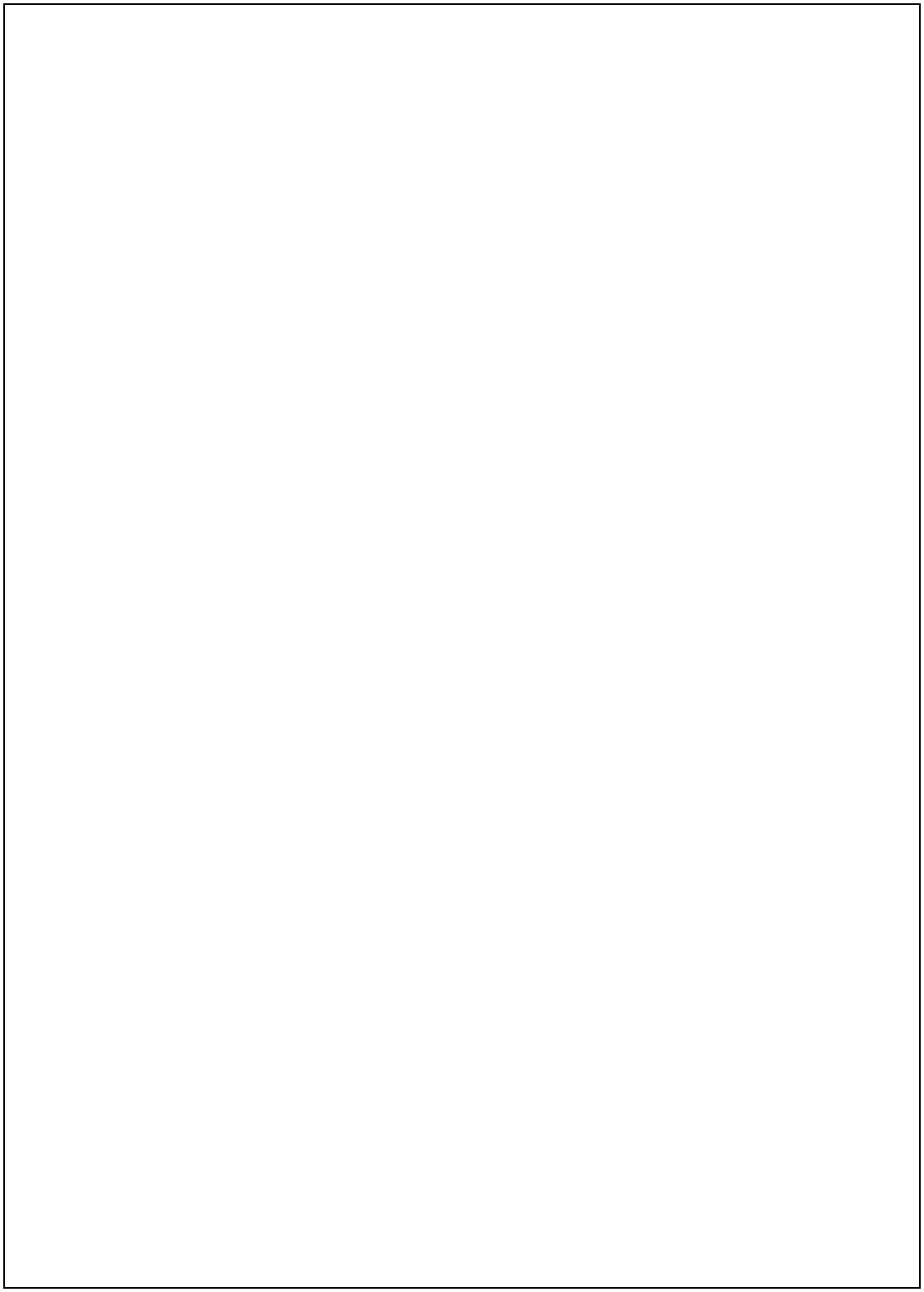
APPENDIX B

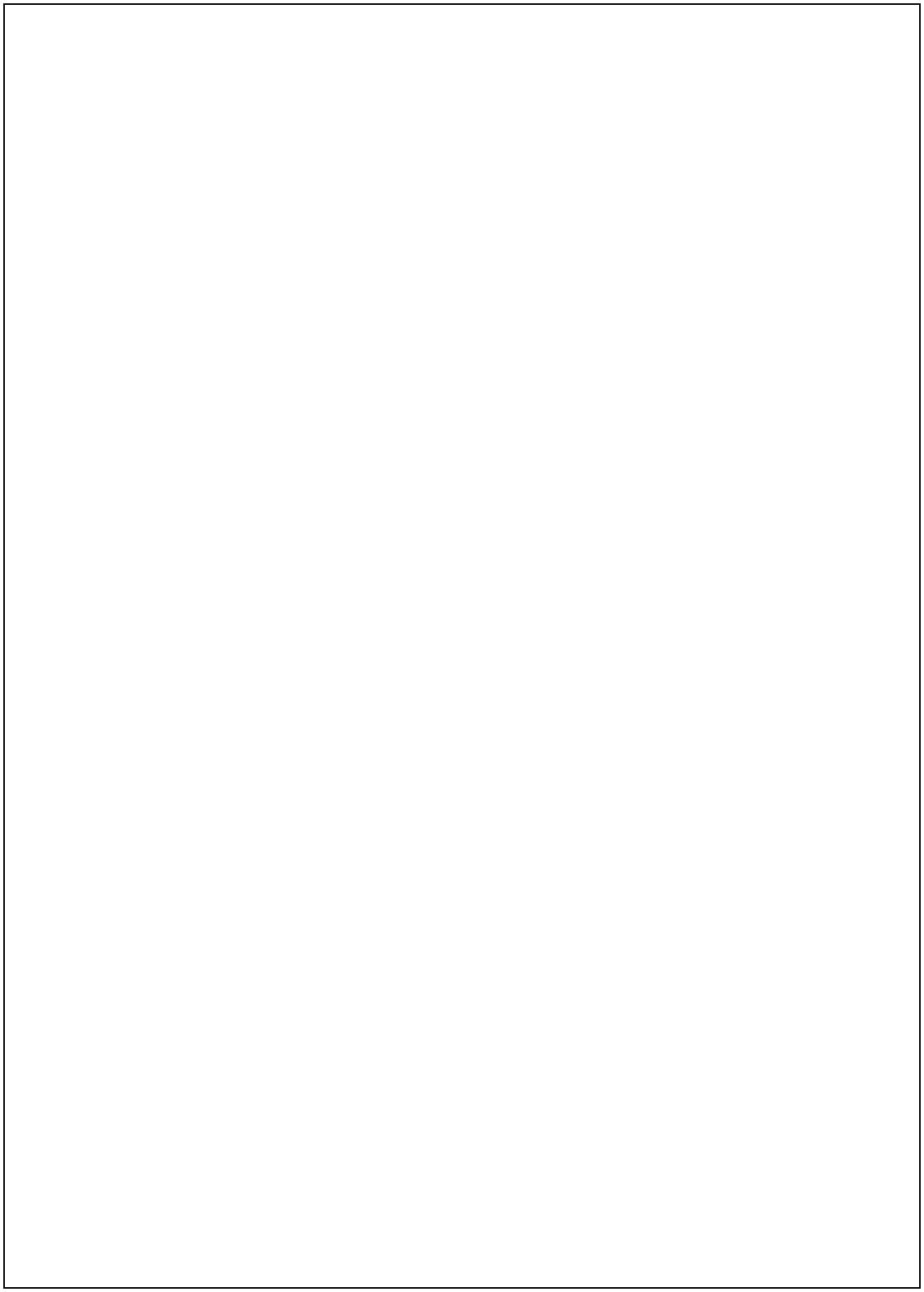
QUESTIONNAIRE

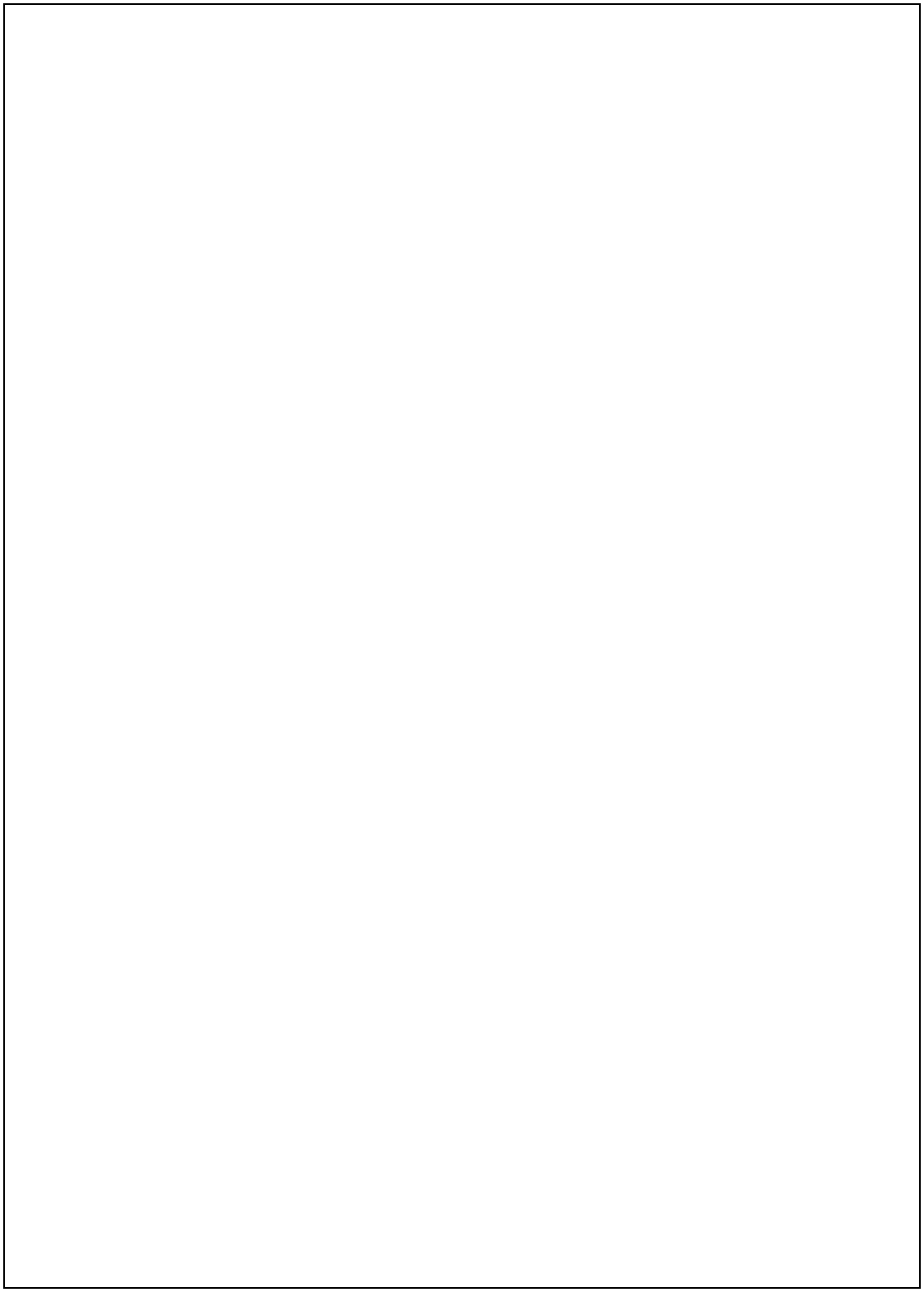








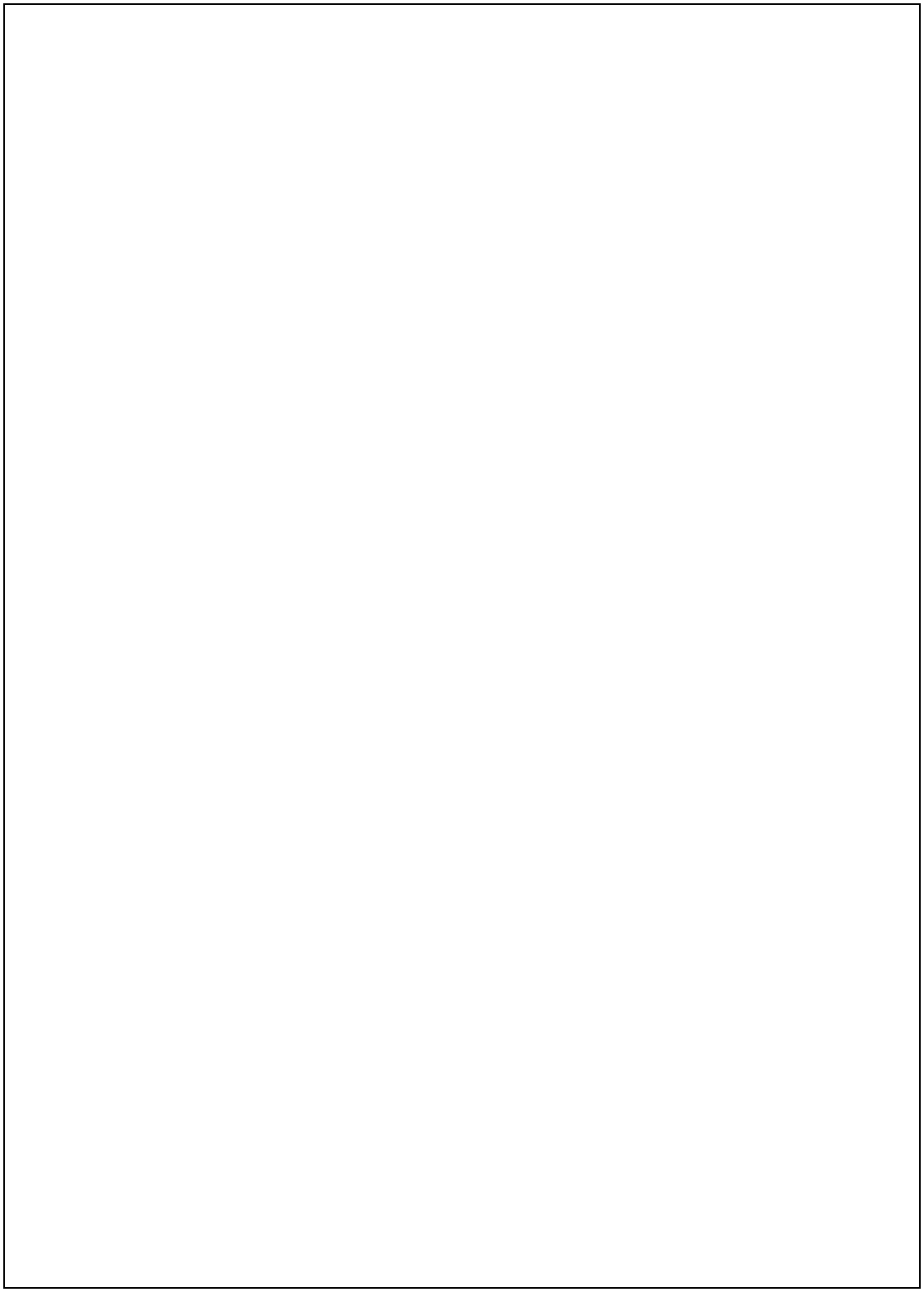


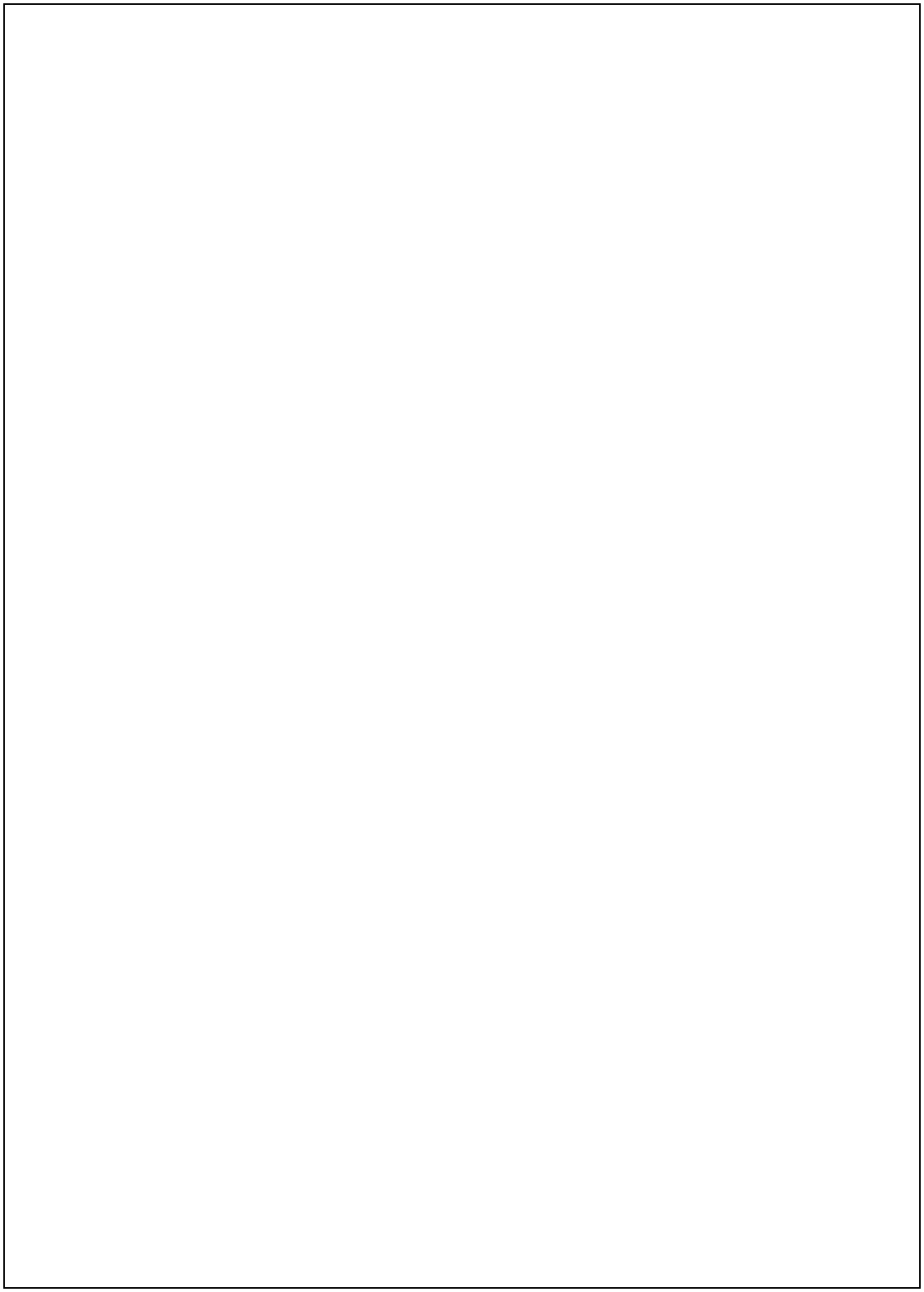


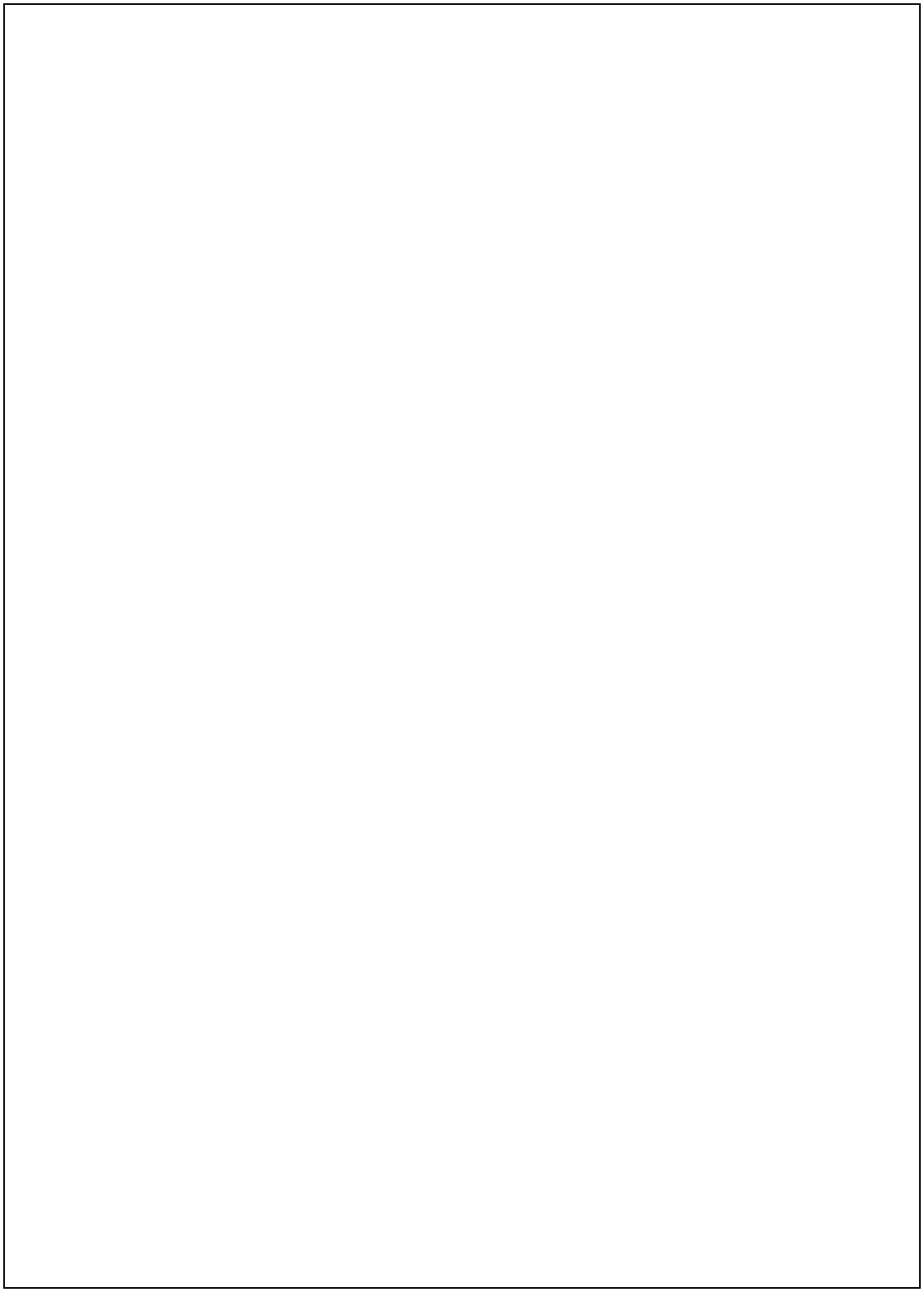
APPENDIX C

PROTOCOL SHEETS

(SELECTION)







APPENDIX D

VIDEO EVALUATION SHEET

Group: _____
Subtask: _____
Time: ☐

	White	Colors
grid structure chaos		

Agreement on coordinate system? ☐ Yes ☐ No Correct? ☐ Yes ☐ No
What kind? _____

	first cable	second cable	third cable	fourth cable
1. Taking Cable initiated by using	☐ w ☐ y ☐ b ☐ g ☐ wearable ☐ desktop ☐ color ☐ pointing	☐ w ☐ y ☐ b ☐ g ☐ wearable ☐ desktop ☐ color ☐ pointing	☐ w ☐ y ☐ b ☐ g ☐ wearable ☐ desktop ☐ color ☐ pointing	☐ w ☐ y ☐ b ☐ g ☐ wearable ☐ desktop ☐ color ☐ pointing
2. First/Second Hole	☐ ☐ **absolute** ☐ ☐ relative ☐ ☐ approximately ☐ ☐ referential → voice	☐ ☐ **absolute** ☐ ☐ relative ☐ ☐ approximately ☐ ☐ referential → voice	☐ ☐ **absolute** ☐ ☐ relative ☐ ☐ approximately ☐ ☐ referential → voice	☐ ☐ **absolute** ☐ ☐ relative ☐ ☐ approximately ☐ ☐ referential → voice
	☐ ☐ **direct** ☐ ☐ indicating direction ☐ ☐ correcting → pointing	☐ ☐ **direct** ☐ ☐ indicating direction ☐ ☐ correcting → pointing	☐ ☐ **direct** ☐ ☐ indicating direction ☐ ☐ correcting → pointing	☐ ☐ **direct** ☐ ☐ indicating direction ☐ ☐ correcting → pointing
	☐ ☐ short ☐ ☐ long ☐ ☐ permanent → freezing	☐ ☐ short ☐ ☐ long ☐ ☐ permanent → freezing	☐ ☐ short ☐ ☐ long ☐ ☐ permanent → freezing	☐ ☐ short ☐ ☐ long ☐ ☐ permanent → freezing
decisive element	☐ ☐ voice ☐ ☐ pointing ☐ ☐ can't tell	☐ ☐ voice ☐ ☐ pointing ☐ ☐ can't tell	☐ ☐ voice ☐ ☐ pointing ☐ ☐ can't tell	☐ ☐ voice ☐ ☐ pointing ☐ ☐ can't tell

3. Dialog

Initiator D/W

	Wearable	Desktop
acknowledge		
confirm. quest.		
other quest.		
meta comm.		
techn. problem		
disagreement		
camera position		

4. Problems

☐ colors of cable
☐ video lag
☐ pointer off
☐ other: _____

APPENDIX E

DATA FROM THE VIDEO

EVALUATION

TABLE 4. Decisive Element for the Different Subtasks Given As Percentages

	voice	pointing	undecidable
Subtask 1&2	100	0	0
Subtask 3&4	0.694444	97.22222	2.083333
Subtask 5-12	3.645833	90.45139	5.902778
All Subtasks	19.21296	76.50463	4.282407

TABLE 5. Number of Pointing and Freezing Actions per Group. For Freezing the Results for Short, Long and Permanent Freezing Are Shown.

	Pointing Actions	Freezing Actions	short	long	Permanent
Group 1	61	31	2	29	0
Group 2	64	62	1	9	52
Group 3	64	0	0	0	0
Group 4	64	2	0	2	0
Group 5	64	52	1	50	1
Group 6	63	41	7	35	0
Group 7	64	63	0	37	26
Group 8	64	26	1	25	0
Group 9	64	50	9	41	0

TABLE 6. Number of Utterances That Fall Into the Different Categories and Number of Pointing Actions for Each Type of Template (* Grid and Structure each only occur twice in Tasks 5-12, therefore we multiplied their value by two to get comparable results.)

	absolute	relative	approximately	referential	pointing
grid*	102	16	0	112	284
structure*	94	22	2	106	286
chaos	51	6	2	161	286

BIBLIOGRAPHY

1. Bauer, Martin; Heiber, Timo; Kortuem, Gerd, and Segall, Zary. A Collaborative Wearable System with Remote Sensing. Proceedings of the Second International Symposium on Wearable Computers; 1998 Oct; Pittsburgh, PA. Los Alamitos, CA: IEEE Computer Society; 1998; c1998: pp.10-17.

2. Billinghurst, M; Bowskill, M.; Jessop, M., and Morphett, J. A Wearable Spatial Conferencing Space. Proceedings of the Second International Symposium on Wearable Computers; 1998 Oct; Pittsburgh, PA. Los Alamitos, CA: IEEE Computer Society; 1998; c1998: pp. 76-83.

3. Crowley, Terrence; Milazzo, Paul; Baker, Ellie; Forsdick, Harry, and Tomlinson, Raymond. MMConf: An Infrastructure for Building Shared Multimedia Applications. Proceedings of Computer Supported Cooperative Work (CSCW) 1990; 1990 Oct: ACM; 1990: pp. 329-342.

4. Dr. Dobbs Journal. The Component Object Model: Technical Overview; 1994 Dec; 1994 http://www.microsoft.com/com/wpaper/Com_modl.asp.

5. Esposito, Chris (Boeing). personal communication; 1998.

6. Gaver, Wiliam; Sellen, Abigail; Heath, Christian, and Luff, Paul. One is not enough: multiple views in a media space. Proceedings of INTERCHI 93; 1993: ACM; c1993: pp. 335-341.

7. Greenberg, Saul and Roseman, Mark. GroubWeb: A WWW Browser as Real Time Groupware. In Companion Proceedings of the ACM SIGCHI'96 Conference on Human Factors in Computing Systems; 1996: ACM Press; 1996; c1996: pp. 271-272.

8. Handkey Corporation. Twiddler; 1998 http://www.handykey.com/.

9. i-O Display Systems. i-glasses!; 1998 http://www.i-glasses.com/.

10. Kortuem, Gerd (University of Oregon). Some Issues in the Design of User-Interfaces for Collaborative Wearable Computers. VRAIS 1998; 1998; c1998 http://www.hitl.washington.edu/people/grof/VRAIS98/Kortuem.html.

11. Kortuem, Gerd; Segall, Zary, and Bauer, Martin. Context-Aware, Adaptive Wearable Computers as Remote Interfaces to 'Intelligent' Environments. Proceedings of the Second International Conference on Wearable Computers ; 1998; Pittsburgh, PA. Los Alamitos, CA: IEEE Computer Society; 1998; c1998: pp. 58-65.

12. Kortuem, Gerd; Segall, Zary, and Bauer, Martin. NETMAN: The Design of a Collaborative Wearable Computer System. MONET: Mobile Networks and Applications; to appear.

13. Krassel, Alex and Robinson, Steve. Microsoft's NetMeeting 2.1 COM Interfaces: Understanding how they work; 1998 Apr: Panther Software; 1998 http://www.microsoft.com/workshop/messaging/netmtg/netmtgcom.asp.

14. Kraut, Robert E.; Miller, Mark D., and Siegel, Jane. Collaboration in Performance of Physical Tasks: Effects on Outcomes and Communication. Computer Supported Cooperative Work '96 Cambridge, MA, USA: ACM; 1996.

15. Kuzuoka, Hideaki; Kosuge, Toshio, and Tanaka, Masatomo. GestureCam: A Video Communication System for Sympathetic Remote Collaboration. CSCW '94; 1994 Chapel Hill, NC, USA: ACM; 1994; c1994: pp. 35-43.

16. Mann, Steve. Definition of Wearable Computing; 1998 http://www.wearcomp.org/def.html.

17. Mann, Steve. Wearable Computing as Means for Personal Empowerment. Keynote Address for the International Conference on Wearable Computing. Fairfax, VA; 1998.

18. Mantei, Marilyn M.; Baecker, Ronald M.; Sellen, Abigail J.; Buxton, William A. S., and Milligan, Thomas. Experiences in the Use of a Media Space. Proceedings of CHI'91 Human Factors in Computing Systems; 1991: pp. 203-209.

19. Metricom, Inc. Ricochet Modem; 1998 http://www.ricochet.net/.

20. Microsoft Corporation. NetMeeting; 1998 http://www.microsoft.com/netmeeting/.

21. Microsoft Corporation. NetMeeting FAQ; c1997 http://msdn.microsoft.com/developer/sdk/netmeeting/netm0056.htm.

22. Microsoft Corporation. NetMeeting Online Documentation; 1997 http://msdn.microsoft.com/developer/sdk/netmeeting/default.htm.

23. Nogatech. Nogatech ClipCam; c1996 http://www.nogatech.com/.

24. Rhodes, Bradley (MIT Media Lab). The Wearable Remembrance Agent: A System for Augmented Memory. Proceedings of the First International Conference on Wearable Computers; 1997 Oct; Cambridge, MA. Los Alamitos, CA: IEEE Computer Society; 1997; c1997: pp. 123-128.

25. Starner, Thad; Mann, Steve; Rhodes, Bradley; Levine, Jeffrey; Healey, Jennifer; Kirsch, Dana; Picard, Rosalind W., and Pentland, Alex (MIT Media Lab). Augmented Reality Through Wearable Computing. M.I.T Media Laboratory Perceptual Computing Section Technical Report No. 397Cambridge, MA: Massachusetts Institute of Technology; 1997; c1997.

26. Stefik, Mark; Foster, Gregg; Bobrow, Daniel G.; Kahn, Kenneth; Laning, Stan, and Suchman, Lucy. Beyond the Chalkboard: Computer Support for Collaboration and Problem Solving in Meetings. Computer Supported Cooperative Work: A Book of Readings Edited by Irene GreifSan Mateo, CA: Morgan Kaufmann Publishers, Inc.; 1988; c1988: pp. 335-366.

27. Stein, Robert; Ferrero, Stephen; Hetfield, Margaret; Quinn, Alan, and Krichever, Mark. Development of a Commercially Successful Wearable Data Collection System. Proceedings of the Second International Symposium on Wearable Computing; 1998; Pittsburgh. Los Alamitos, CA: IEEE Computer Society; 1998; c1998: p. 18-24.

28. Tang, John C.; Isaacs, Ellen A., and Rua, Monica (SunSoft Inc.). Supporting Distributed Groups with a Montage of Lightweight Interactions. Proceedings of Computer Supported Cooperative Work 1994; 1994Chapel Hill, NC: ACM; 1994.

29. ViA Inc. ViA Wearable Computer; 1998 http://www.flexipc.com .

30. Whittaker, Steve (Lotus Development Corporation). Video As A Technology For Interpersonal Communications: A New Perspective. Proceedings of Multimedia Computing and Networking 1995; 1995: SPIE - The International Society for Optical Engineering; 1995: pp. 294-304.