

EyeTap reality mediators, wearable computers and wireless  
networks for interactive and shared mediated reality systems

by

Felix Tang

A thesis submitted in conformity with the requirements  
for the degree of Master of Applied Science  
Department of Electrical and Computer Engineering  
University of Toronto

Copyright by Felix Tang, 2002

# EyeTap reality mediators, wearable computers and wireless networks for interactive and shared mediated reality systems

Felix Tang

Master of Applied Science

2002

Department of Electrical and Computer Engineering

University of Toronto

## Abstract

“Mediated reality affords the EyeTap apparatus the ability to augment, diminish or otherwise alter our perception of reality.”[1] The goal of this research is to demonstrate interactive and shared mediated realities.

The EyeTap reality mediator is used with Bluetooth, 802.11b and IrDA wireless technologies to implement a system that allows for the interaction between the wearer and objects in the environment. The EyeTap device can mediate reality with information such as labels and graphical user interfaces that is transmitted over wireless networks from objects in the environment. A shared mediated reality is demonstrated through the implementation of a system that allows for the mapping of an EyeTap wearer’s environment through the VideoOrbits Gyroscopic Head–Tracker. The environment map is then transmitted to another EyeTap wearer, who, through head motion is able to browse the remote environment as if it were their own.

## Acknowledgements

I would like to thank Steve Mann for teaching me that “later means never”, how doing is better than just talking and showing me that the word “impossible” is really just a figment of my imagination although I’m still trying to wrap my mind around all these *ideas*.

I would like to thank James Fung and Corey Manders for their help, support and insatiable gastronomic appetites. I would also like to thank Chris Aimone for teaching me to walk strangely and jump on rocks. Finally, to Jean Tourrilhes who helped me write my first Linux kernel driver and for working on almost every free/open source project I have been interested in.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Background</b>	<b>3</b>
2.1	Mediated Reality and EyeTap devices . . . . .	3
2.2	Humanistic Intelligence . . . . .	6
<b>3</b>	<b>Implementing an EyeTap System</b>	<b>9</b>
3.1	EyeTap and WearComp . . . . .	9
3.2	GNU/Linux . . . . .	11
3.2.1	IEEE 1394 . . . . .	11
3.2.2	Wireless networking: Bluetooth, 802.11b, IrDA . . . . .	11
3.3	xcaplive: a reality processing program . . . . .	16
<b>4</b>	<b>AMRE: Active Mediated Reality Environments</b>	<b>19</b>
4.1	Mediated reality and interacting with the world. . . . .	19
4.2	Active Mediated Reality Environments . . . . .	20
4.2.1	Implementation . . . . .	21
4.3	Mediated Reality . . . . .	23
4.4	Summary . . . . .	23
<b>5</b>	<b>Hardware Accelerated VideoOrbits</b>	<b>25</b>
5.1	Background . . . . .	25
5.1.1	VideoOrbits and Projective Flow . . . . .	25
5.1.2	VideoOrbits: Image Registration and Image Composites . . . . .	28
5.2	Optimizing VideoOrbits . . . . .	28

5.3	Mapping Projective Coordinate Transformations . . . . .	30
5.4	Measurements of OpenGL acceleration . . . . .	32
5.5	Summary . . . . .	36
<b>6</b>	<b>Seeing Eye to Eye</b>	<b>37</b>
6.1	Background . . . . .	38
6.1.1	VOHT: Video Orbits Head Tracking . . . . .	38
6.1.2	VOGHT: Video Orbits Gyroscopic Head Tracker . . . . .	40
6.2	Shared Mediated Reality . . . . .	44
6.2.1	VideoOrbits . . . . .	45
6.2.2	Head Tracking . . . . .	46
6.2.3	VOGHT: Reference Frame Database . . . . .	46
6.3	Seeing Eye to Eye: a shared mediated reality . . . . .	47
6.3.1	Demonstration . . . . .	52
6.4	Accounting for the Camera Intrinsic Parameters . . . . .	52
6.5	Summary . . . . .	53
<b>7</b>	<b>Conclusion</b>	<b>55</b>

# List of Figures

2.1	Virtual, Augmented and Mediated reality. . . . .	4
2.2	EyeTap device . . . . .	5
2.3	Signal flow paths in Humanistic Intelligence . . . . .	8
3.1	EyeTap device . . . . .	10
3.2	802.11b: Infrastructure mode . . . . .	13
3.3	802.11b: Ad-hoc mode . . . . .	13
3.4	802.11b: Ad-hoc mode consisting of mobile nodes . . . . .	13
3.5	An IrDA transceiver designed and built by the author. . . . .	15
3.6	xcaplive: program design . . . . .	17
4.1	Mediated reality view of a Laptop . . . . .	21
4.2	Mediated reality view through an EyeTap . . . . .	22
4.3	RUI with a GUI placement . . . . .	23
5.1	VideoOrbits in action . . . . .	29
5.2	VideoOrbits generated image composite . . . . .	30
5.3	Hardware accelerated VideoOrbits algorithm . . . . .	31
5.4	Projection times for VideoOrbits . . . . .	33
5.5	Projection times for VideoOrbits, varying image sizes . . . . .	34
5.6	Projection times for VideoOrbits, varying projection parameters . . . . .	35
6.1	Composing projective coordinate transforms . . . . .	39
6.2	System Drift With and Without Reference Frames . . . . .	39
6.3	VideoOrbits head-tracking algorithm . . . . .	41
6.4	VideoOrbits vs Gyroscope Tracking . . . . .	42

6.5	Tracking System Block Diagram . . . . .	43
6.6	EyeTap wearer A, EyeTap wearer B. . . . .	44
6.7	Outdoors: single reference frame Environment map . . . . .	48
6.8	Outdoors: multiple reference frame Environment map . . . . .	49
6.9	Indoors: generating an environment map . . . . .	50

# List of Tables

3.1	Comparison of USB, IEEE1394 and PCI Frame Grabber for image capture. . . . .	10
-----	--	----



# Chapter 1

## Introduction

The use of computers has been integrated into daily life, especially with the prevalence of the internet. This demand for information and the necessary computer machinery to access it is demonstrated by the widespread acceptance of personal computing devices such as cellular phones and personal digital assistants.

These devices are limited in their features and represent an attempt to address the issues of mobile computing. The field of wearable computing presents a different solution to the integration of the computer and access to information. Through wearable computing and mediated reality, new methods for interacting with the environment are created. Specifically, mediated reality involves changing the way a user views and interacts with the world. This is done through the use of an EyeTap reality mediator which “taps” the eye by functioning as both a camera and display. The display is the inverse of the camera and is referred to as the “aremac” [1]. The EyeTap device allows for the capture, transmission and display of “live” video. This video or sequence of frames of images can be computationally processed to create a mediated reality.

The design and implementation of an EyeTap, wearable computer and wireless networks is discussed in Chapter 3. The wireless EyeTap system is used to demonstrate active mediated reality environments (AMRE). An AMRE system has objects that provide information and interfaces available only through the EyeTap reality mediator as described in Chapter 4.

The ability to mathematically relate separate frames of images from a video sequence is done through the use of the VideoOrbits algorithm. The VideoOrbits algorithm calculates the exact projective coordinate transformation that registers successive pairs of images in a sequence. The VideoOrbits algorithm allows for the creation of image composites, calculation of egomotion (i.e.

head tracking) and ad-replacement (an example of mediated reality).

For the goal of a fully mediated reality, the VideoOrbits algorithm will need to run in real-time. Since the VideoOrbits algorithm and its implementation are used as the basis of much of the research, increasing the speed of VideoOrbits is desirable. Through the use of existing computer graphics hardware and the OpenGL standard, sections of the VideoOrbits algorithm were optimized. This takes computer graphics hardware which is generally used for image synthesis and instead uses it for accelerating a computer vision algorithm. This is discussed in Chapter 5.

The concept of a shared mediated reality is explored in Chapter 6. A wireless network of wearable computer users are able to map and share their current environments. The remote user is able to browse the environment map solely through head motions and the images generated correspond to the perspective of the remote users head motion.

## **Chapter 2**

# **Background: Mediated Reality, EyeTap devices and Humanistic Intelligence**

### **2.1 Virtual, Augmented, and Mediated Reality with wearable computers and EyeTap devices**

The need for information and the use of computers in accessing that information has always been limited by the location of the computing machinery. From mainframes to personal computers and laptops, information was only accessible while the user was interacting with the machine. The wearable computer solves this problem by making the computing machinery available to the user at all times. In putting the computing machinery on the person's body, issues such as weight, bulkiness and usability are important. But by moving away from the desktop paradigm, human interactions with the machine have fundamentally changed.

The development of the wearable computer has been facilitated by the use of the head mounted display (HMD) [2]. In [2] the input to the display was entirely computer generated and this resulted in "virtual reality". In a virtual reality the user only sees rays of virtual light that are generated by a computer.

HMDs with see-through displays or HMDs with video feeds from the real world have been used to superimpose computer graphics on the real world, resulting in "augmented reality". However, registration of the real world and the computer generated information is complex and has involved the use of laser range finders and head trackers [3]. In an augmented reality, the user

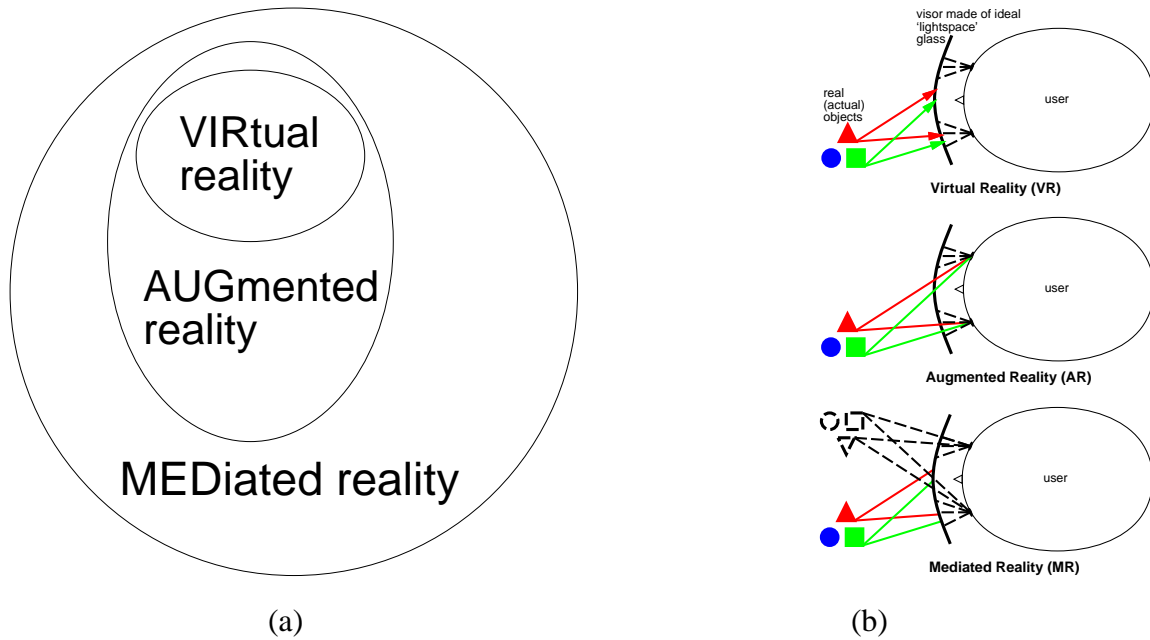


Figure 2.1: (a) Relationship of Virtual, Augmented and Mediated reality. (b) The relationship between synthetic and real light in Virtual, Augmented and Mediated Realities.

views both virtual light and real light or a regenerated version of the real light.

The development of virtual and augmented realities have been done with HMDs that are connected via cable to a computer. The user is able to move around in a small space but is not completely mobile.

The use of a wearable computer with a reality mediator alters the way a person interacts with the world. A reality mediator such as the EyeTap device allows the wearable computer user to interact with the computer and the environment in a constant manner. There are two types of constancy:

1. Operational constancy
2. Interactional constancy

This is in contrast to handheld video recorders, personal digital assistants and laptops. The handheld video recorder needs to be turned on and as such lacks operational constancy. Even if it were to be left on, it lacks interactional constancy because it takes time to actually use the

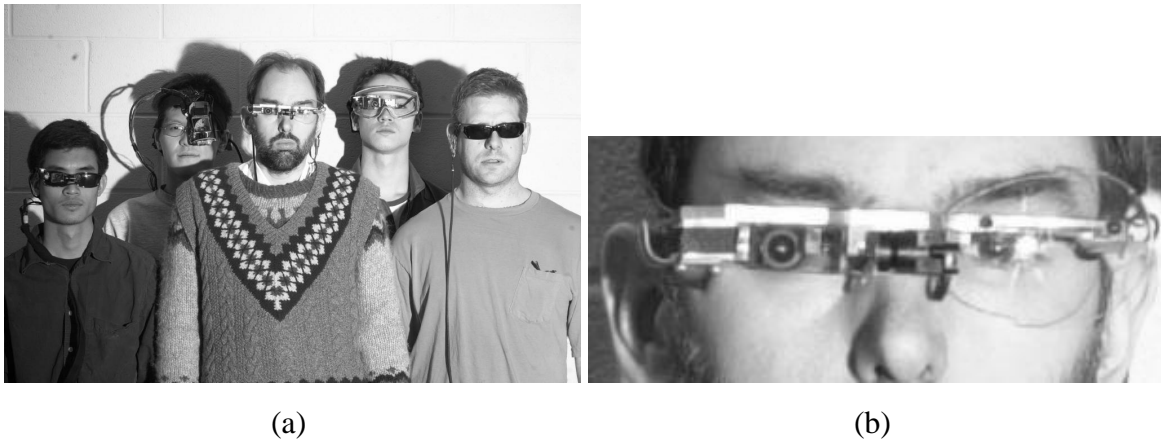


Figure 2.2: (a) Examples of various different kinds of EyeTap devices. (b) Close up of an EyeTap device. Notice how the right eye is tapped at the center of projection and appears to be replaced by the camera.

viewfinder and start recording. If it were to be held up to the eye all the time then it would have interactional constancy which implies operational constancy.

The EyeTap device, however, embodies operational and interactional constancy. It consists of three main parts:

1. A digital camera that measures and digitizes the incoming rays of light
2. A digital system to process this digitized stream of information
3. An output device that converts a stream of numbers into rays of light

The system for processing the information is generally a wearable computer (WearComp). The system as described, allows for incoming rays of light to be digitized and then regenerated collinearly. This device, when placed at the center of projection of the eye, allows for the “tapping” of the incoming rays of light.

When the system includes a wearable computer to process the stream of information, the computer can add, remove, or otherwise modify the images before they are presented to the user. This results in a computer mediated reality. It is emphasized that the real world can either be augmented with computer generated information, or real world objects can be replaced and diminished by computer generated information. This is in contrast to augmented reality in which computer generated

information can only be added to the scene, but the occlusion or removal of real world information is not done due to the inability to completely block out the real source except in certain special cases[3].

If the incoming stream of data is not processed, then upon the regeneration of the light, the image is not changed in anyway, barring of course, technical limitations of the implemented EyeTap device (e.g. monochrome display).

Tapping the highest bandwidth signal source of the wearer with an EyeTap and computationally processing this information creates a “mediated reality”.

## 2.2 Humanistic Intelligence

Humanistic Intelligence is the result of the introduction of the human into the computational feedback loop. Through the “operational constancy” of the wearable computer and the “interactional constancy” afforded through the reality mediator, this humanistic intelligence is developed. The computer runs in a supporting role by performing continuous tasks such as monitoring sensors or processing video, while the human provides the “intelligence”; the decision making ability which is complicated and difficult to reproduce in a machine.

The prevalence of microcontrollers and wireless devices have promoted the concept of “ubiquitous computing” or “smart rooms”. These are environments in which there are many sensors (e.g. cameras, speakers, motion detectors, etc) and a central system that attempts to predict what the user is doing in order to assist the user. The reality is that it is quite difficult to understand what a user’s true intentions are. For instance, the intelligent lights in a room which are activated by motion sensors turn off when the user is actively doing work in their office because they are no longer moving about.

However, the inverse of that system would make the controls to the environment available to the wearable computer user. In fact it would replace the “smart” aspect of the system with the users “humanistic intelligence” and through the wearable computer provide the method for controlling the environment; in effect, replacing artificial intelligence with humanistic intelligence. For example, the user, through configuring the wearable computer, could make known their preferences for lighting, volume and temperature and the wearable computer would adjust the environment as needed. This would take into account individual preference, which a smart room would have a

difficult time determining, barring sophisticated facial recognition or the use of tags with unique identifiers.

The difference is in who actually has control over the room. A person in a “smart room” is controlled by someone somewhere else or by no one at all; the person is at the mercy of the machine. A person who can control a room by interacting with it, is in charge of the decision making and the room is just responding to the persons commands.

The computer is in a supporting role, one in which it assists the wearer in completing day to day activities (refer to Figure 2.3). Humanistic intelligence is important in the motivation it gives for the implementation of the research in Chapters 4 and 6.

The author has used the wearable computer to present a list of wireless devices that are viewable through the EyeTap. This is a filtered version of the available devices in the room defining the context of what is interesting as those devices that are currently viewable. This context is necessary because there are many devices to interact with and it could result in an overload of information. Of course, the system could be extended to present a listing of all devices in a familiar environment. This allows the wearer to use devices that are not currently viewable but of interest. The computer is functioning in a supporting role to enhance the user’s primary task of interacting with the environment.

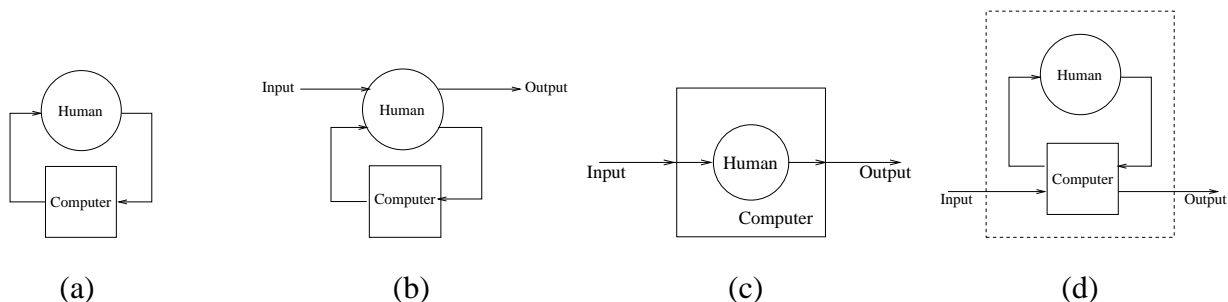


Figure 2.3: The three basic operational modes of WearComp. (a) Signal flow paths for a computer system that runs continuously, constantly attentive to the user’s input, and constantly providing information to the user. Over time, constancy leads to a symbiosis in which the user and computer become part of each other’s feedback loops. (b) Signal flow path for augmented intelligence and augmented reality. Interaction with the computer is secondary to another primary activity, such as walking, attending a meeting, or perhaps doing something that requires full hand-to eye coordination, like running down stairs or playing volleyball. Because the other primary activity is often one that requires the human to be attentive to the environment as well as unencumbered, the computer must be able to operate in the background to augment the primary experience, for example, by providing a map of a building interior, or providing other information, through the use of computer graphics overlays superimposed on top of the real world. (c) The wearable computer can be used like clothing, to encapsulate the user and function as a protective shell, whether to protect us from cold, protect us from physical attack (as traditionally facilitated by armour), or to provide privacy (by concealing personal information and personal attributes from others). In terms of signal flow, this encapsulation facilitates the possible mediation of incoming information to permit solitude, and the possible mediation of outgoing information to permit privacy. It is not so much the absolute blocking of these information channels that is important; it is the fact that the wearer can control to what extent, and when, these channels are blocked, modified, attenuated, or amplified, in various degrees, that makes wearable computing much more empowering to the user than other similar forms of portable computing. (d) An equivalent depiction of encapsulation (mediation) redrawn to depict the interaction between user and computer, where the encapsulation is understood to comprise a separate protective shell. This diagram is from[1].

## Chapter 3

# Implementing an EyeTap and WearComp for a mediated reality system

“The terms systems design, systems engineering, and systems design engineering all refer to the same intellectual process of being able to define and model complex interactions among many components that comprise a natural system (such as an ecosystem and human settlement) or artificial system (such as a spacecraft or intelligent robot), and being able to implement the system with proper and effective use of available resources.”[4]

The implementation of an EyeTap reality mediator comprising of an EyeTap device, wearable computer (WearComp) and wireless networking technologies forms the basis of the system that is used in the research presented in Chapters 4 and 6.

### 3.1 EyeTap and WearComp

The EyeTap device consists of a camera and an aremac separated by a diverter. The diverter functions by diverting inbound light into the camera so that it may be computationally processed and then resynthesized by the aremac, this is illustrated in Figure 3.1 (a). The diverter effectively places the center of projection of the camera at the center of projection of the eye, the eyetap point. The aremac is a computer controlled light synthesizer that is used to draw on the user’s retina.

The author’s right EyeTap (Figure 3.1 (b)) uses a fully opaque double-sided mirror which

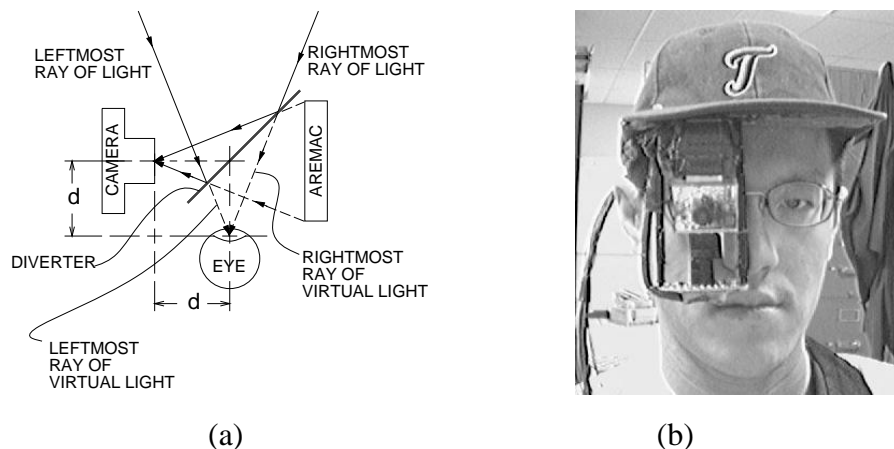


Figure 3.1: (a) Schematic drawing of an EyeTap device. The center of projection of the eye and the camera are equidistant from the diverter, this distance “ $d$ ”, is called the eyetap distance. (b) The authors EyeTap device.

allows the computer to augment and/or diminish portions of the image before presenting it to the user. This is in contrast to a partially silvered mirror which allows for an augmented reality but not a diminished reality except in special cases where the virtual light is much brighter than light from the real world. The aremac is a 640x480, green monochrome Kopin microdisplay from an early generation Xybernaut.

The camera used is an IEEE1394 progressive scan digital camera by ADS Technologies. A IEEE1394 camera was chosen based on the following information:

	Available on WearComp	Cameras Available	Capture Ability
USB	Yes	Yes, low quality.	320x240, 8 fps
IEEE1394	Yes	Yes, low-high quality.	640x480, 30 fps
FRAME GRABBER	No	Yes, low-high quality.	640x480, 30fps

Table 3.1: Comparison of USB, IEEE1394 and PCI Frame Grabber for image capture.

USB is a 12 Mbps low-speed bus that is physically unable to handle uncompressed data at the sizes and frame rates desired. Frame grabbers are PCI expansion cards, that allow general purpose computers to do NTSC/PAL/SECAM video capture. Due to the fact that they are expansion cards they are not useful in compact wearable computer systems. The author’s work with IEEE 1394 is explained in Section 3.2.1.

The WearComp is a Sony Vaio Z505LS with a 750 MHz Pentium Pro PIII processor, 128 MB of RAM, 20 GB harddrive, IEEE1394, USB, IrDA, and PCMCIA.

## **3.2 GNU/Linux**

The wearable computer system uses the GNU/Linux operating system. This choice was made based on the availability of source code and the ability to do research in a manner that would not be restricted by an end user license agreement (EULA). The source code allows for the rapid understanding of complex operating system and programmatic concepts and eases the development of programs that are necessary in the adaptation of a general purpose operating system into a mediated reality wearable computing system.

### **3.2.1 IEEE 1394**

IEEE1394 is a high speed serial bus (up to 400 Mbps). It is designed to be a high-speed and low cost method of connecting computer peripherals and other electronic devices. There are many digital cameras that are able to capture progressive scan images in a variety of formats and frame rates. They are typically of much higher quality than similarly priced USB cameras and are simpler to integrate into a computer system than frame grabbers. It was for these reasons that the author explored the use of IEEE 1394 digital cameras in the EyeTap device.

The author was involved in the testing and debugging of IEEE 1394 development in the GNU/Linux operating system. The author also was involved in the testing and debugging of “Co-riander” [5] the first program to use IEEE 1394 digital cameras in GNU/Linux.

### **3.2.2 Wireless networking: Bluetooth, 802.11b, IrDA**

The use of wireless networking allows the wearable computer to transmit information and interact with other devices. The following wireless networking technologies were explored based on the fact that they were readily available and in common use. Wireless technologies such as packet radio were not examined due to the need for an “Amateur Radio Operator Certificate with Basic Qualification”. The motivation is the use and exploration of technologies that are available to the general public. These wireless technologies are an integral part of the systems presented in

Chapters 4 and 6.

## **Bluetooth**

Bluetooth is a low power spread spectrum frequency hopping wireless technology in the unlicensed 2.4 gigahertz range. It is currently designed to operate at data transfer rates of up to 721 Kbps in half duplex with a range of 10 meters. The current implementation of the Bluetooth driver for GNU/Linux, BlueZ [6] supports hardware such as the Ericsson ROK 101 007 Application and Training Tool Kit and the Xircom CreditCard Bluetooth Adapter which were used to connect wearable computers.

The connections used the point-to-point protocol (PPP) to establish internet protocol (IP) connections that run over the serial link that Bluetooth provides. A separate PPP connection is needed for each wearable computer in the network. This results in increasing complexity of the network as it grows.

The Bluetooth connections can be used for connecting together wearable computers, but the short range is a hindrance in day-to-day use. Bluetooth is designed for linking together several simpler devices to a wearable computer in a personal area network (PAN). This would be useful in eliminating the wires that connect sensors, keyboards and headphones to the computer, although they would not be able to replace the cable connections for video capture or display due to the insufficient bandwidth. Replacement of the wires to the camera and display would make the EyeTap device less obtrusive. The EyeTap device is situated on the head and wires at this location negatively impact social interaction.

The short range and low bandwidth were factors against the daily use of Bluetooth technology for the wearable computers in the lab.

## **IEEE 802.11b**

IEEE 802.11b is a direct-sequence spread spectrum wireless technology in the unlicensed 2.4 gigahertz range. It is a newer technology compared to IEEE 802.11 which operates at a maximum of 2 Mbps. IEEE 802.11b operates at speeds of up to 11 Mbps and will fall back to 5.5 Mbps, 2 Mbps and 1 Mbps depending on signal strength. The range of IEEE 802.11b is approximately 305 metres in open areas and approximately 76-122 metres in closed spaces.

The drivers for the Linksys WUSB ver 2.5 Wireless USB Network Adapter in GNU/Linux

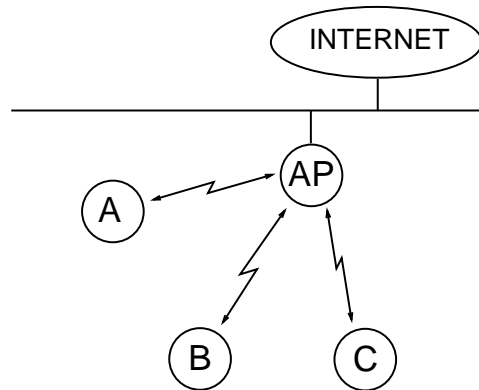


Figure 3.2: 802.11b infrastructure network. If any of the users A, B or C want to communicate with each other they must do so through the AP.

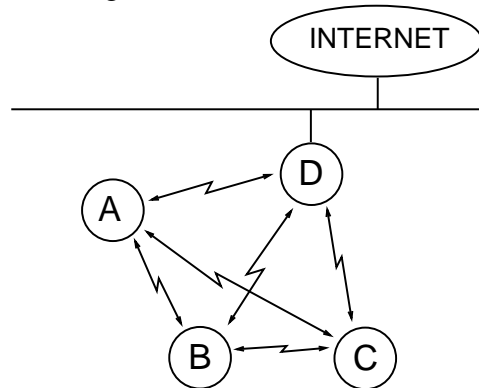


Figure 3.3: 802.11b ad-hoc network. D is a computer which has a wireless and a wired connection. A, B, C and D can communicate directly with each other. A, B, and C can make a connection to the internet through D.

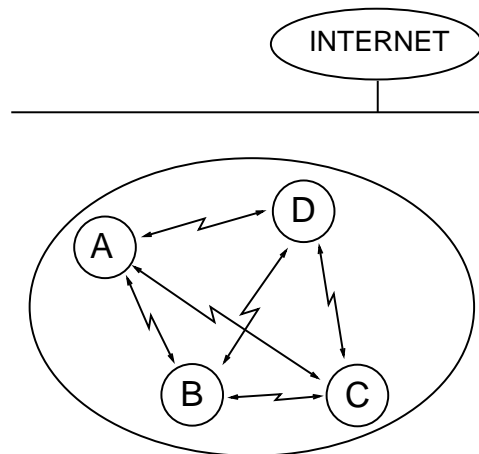


Figure 3.4: 802.11b ad-hoc network with no connection to a wired network. A, B, C and D can communicate directly with each other. There is no connection to the internet in this configuration.

typically allowed for data rates of 250 KBps (2 Mbps). Typically, IEEE 802.11b networks are designed with access points which bridge the wired ethernet networks with the wireless network. These access points have firmware which use dynamic host configuration protocol (DHCP) for automatically assigning IP addresses to the wireless network cards. The 802.11b network cards are then required to operate in infrastructure mode (see Figure 3.2). The University of Toronto Campus Wireless Network (UTORcwn) is an example of this. The author used the GNU/Linux operating system to connect to UTORcwn and made this information available to the University of Toronto Computing and Network Services. This will help others connect to the UTORcwn using GNU/Linux.

The preferred design of a network of wearable computers is that of a peer-to-peer topology. In order to accomplish this, the cards are used in ad-hoc mode and, with the use of network address translation (NAT) and a DHCP server a wireless router (“AP”) was created (refer to Figure 3.3). The main difference between an AP and a router such as the one created, is the ability to use multiple APs to cover a larger area and to automate hand-off of users from one AP to another. It should be noted that APs that are commonly available for the home and home office are not capable of this multiple coverage and hand-off, APs with these capabilities tend to cost substantially more.

This was the type of network created in the laboratory and used daily by the wearable computers. In the operation of a peer-to-peer network, the cards are useful even without a router due to the fact that the cards can communicate directly with each other. This allows the wearable computers to constantly interact with each other even though there is no AP such as off the university campus. This is illustrated in Figure 3.4.

### **IrDA: Infrared Data Association**

IrDA is a wireless technology which works by using infrared light, and it works only when the two devices have a clear line of sight. The line of sight issue can be seen as a feature as well as a drawback. It means that for communications to happen, the devices must be able to “see” each other but at the same time this allows the system to infer that the device is present.

IrDA operates at a range of speeds, serial infrared (SIR) operates at 115Kbps and fast infrared (FIR) operates at 4Mbps. The range on the SIR link is approximately 1-2m as determined experimentally.

An IrDA transceiver is usually built into most wearable computers, but they tend to be on

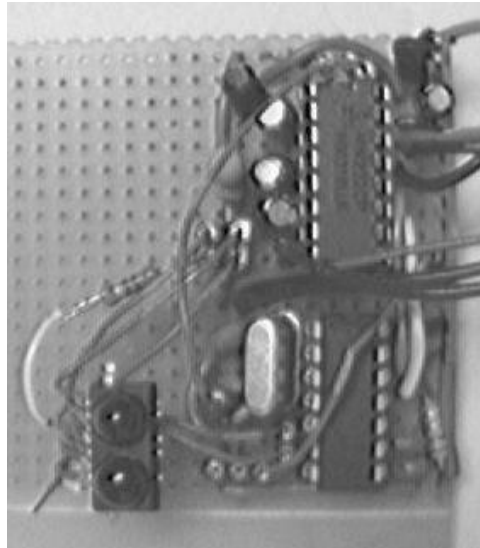


Figure 3.5: An IrDA transceiver designed and built by the author.

the body of the machine. Ideally the IrDA transceiver would be placed on the EyeTap so that interaction with devices by “looking” is possible. To achieve this one might use a commercially available IrDA transceiver, that use serial or USB connectors. Unfortunately they are bulky and tend to be on short cables. This motivated the author to design and prototype a custom IrDA transceiver.

The IrDA transceiver was constructed from a Microchip MCP2120 IrDA encoder decoder chip, MAX232 line driver/receiver, Vishay Telefunken TFDS4500 transceiver and other assorted parts. The schematic and instructions on building the device are available at [7] and is based on a reference design in [8]. The transceiver connects to the wearable computer through a serial cable allowing placement as desired.

This newly built IrDA transceiver did not have an existing Linux kernel driver. A kernel driver is a program that interacts with or controls the hardware and provides a certain set of interfaces to higher level processes in the kernel. Driver programming requires a complete understanding of the hardware that is being controlled (e.g. issues of timing, transfer rates, registers) as well as a thorough understanding of the programmatic interface that is being provided to the kernel and other higher level programs. The author wrote a kernel driver that fit into the existing IrDA infrastructure in the Linux kernel. This driver is now part of the official Linux kernel and is available under the GNU general public license (GPL).

The IrDA transceiver is also controllable by microcontrollers such as the PIC16F877. This allows for the creation of simple and small devices that can use the IrDA technology to wirelessly transfer information.

### 3.3 xcaplive: a reality processing program

`xcaplive` is the program that does the reality mediation processing on the wearable computer. It captures video through the EyeTap camera and displays it through the EyeTap aremac. It is designed for use through a simple keyboard interface and is able to manipulate the frames of incoming video.

The EyeTap devices are custom made prototypes and as such have a variety of configurations of cameras and aremacs. The cameras and aremacs might be rotated  $90^\circ$  or  $180^\circ$  due to design constraints during the construction of the device. Therefore, `xcaplive` is able to take the incoming images and apply a range of transformations such as: rotations of  $90^\circ$  and  $180^\circ$  and flips about the  $x$  and  $y$  axes.

It is also able to record images as they are processed and to transmit them wirelessly. The “live” wireless transmissions of the images allows remote users to view what the user is currently observing. Also, a visual history of the images is available through an internet interface. An example of this is “Seeing Eye People” [9]. Headtracking, gyroscopic headtracking and virtual labels have also been implemented in `xcaplive` though these options require specific hardware to run.

Through working with IEEE 1394 (refer to section 3.2.1), `xcaplive` was reprogrammed to use IEEE 1394 digital cameras. The reprogramming of `xcaplive` incorporated a complete restructuring of the design. Using established software engineering principles in [10] `xcaplive` was redesigned to incorporate the bridge and proxy patterns [11] as well as a source-sink approach to allow for generic inputs and outputs. This allowed for the ease of integration of multiple input sources (e.g. multiple cameras) and multiple output sinks (e.g. live transmission, saving to hard-disk) simultaneously. Multiple input sources allow for the use of stereo computer mediated reality [1].

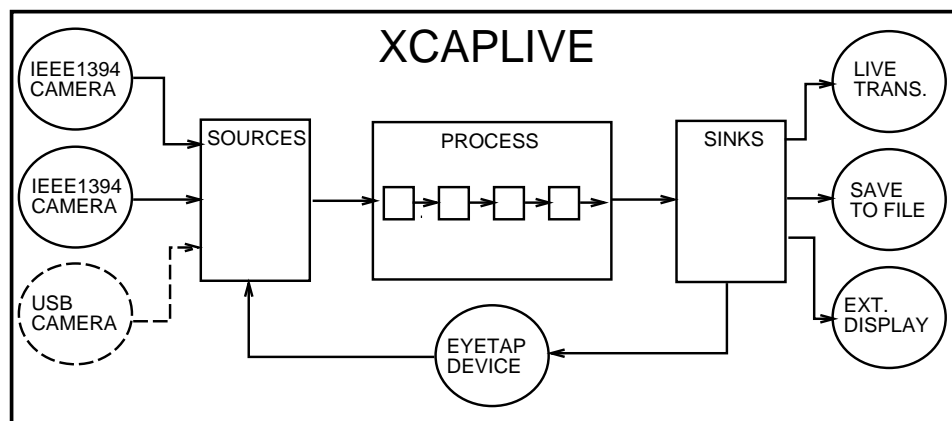


Figure 3.6: `xcaplive` handles sources generically and processes them by applying various filters such as rotations, flips, colour space transformations, head-tracking, etc. Finally, the processed image is then handled by a generic sink which is able to output the image as desired. The source and sink may also be to and from the same device as demonstrated by the EyeTap.



# Chapter 4

## AMRE: Active Mediated Reality Environments

### 4.1 Mediated reality and interacting with the world.

The ubiquitous computing paradigm is described in [12], involves a “smart” aspect which is defined as the many computers and microcontrollers embedded in the room or devices in the environment.

The systems typically function through the use of a beacon which will identify users or objects of interest. This beacon element is typically passive. For the smart room to function, all the sensors have to be networked and linked together to a central system. This system will then process all the inputs and respond. An example is that of heating and lighting; when there is a person in the room, then the program will turn on the lights otherwise they will be turned off to save electricity.

[13] and [14] are examples of a passive beacon with a unique identifier. The identifier is used to both uniquely identify the user and provide access control. This data is broadcast to the system and the allowed actions will be presented or actualized (e.g. ,the locked door will open, music will play). The devices have also been extended to allow for tracking of the user in an office environment.

What if we were to consider the inverse of the system presented? If the beacons or passive elements were distributed about the room (e.g. ,in the light switches, heating and cooling controls) and the wearable computing user carried the “smart” element (i.e. the WearComp), then the control of the system could be based on the user. There would be no need to network the building or have a

central system collecting all the data from the sensors and trying to predict what the user is doing. Instead, the user could interact through the wearable computer with the environment, turning on the lights when the user enters a room and turning off the lights when he/she exits. This puts the system of control on the user in the form of a wearable computer. If the user wished to allow his/her presence to be known, he/she could connect with other users nearby and transmit his/her location.

This is an investigation of work by [15] which is an example of this inverse system. The objects and rooms are embedded with infrared beacons and the receiver or “smart” element is embedded into the user’s EyeTap and WearComp.

## 4.2 Active Mediated Reality Environments

This chapter presents AMRE, a system for the location of, and interaction with, wireless objects through an EyeTap mediated reality environment. In environments containing multiple wireless devices that can be used, there exists the problem of locating and selecting specific devices. AMRE functions by reducing the amount of available wireless devices that are presented to the user only to those currently in the user’s mediated field of view. This is done through the use of a combination of line of sight and non line of sight wireless technologies. Using the wireless technologies to transmit data and control information, the EyeTap device is able to display information such as labels and graphical user interfaces as specified by the remote device. This allows for interaction with the wireless device of interest through the user’s mediated view.

An environment that is densely populated with wireless devices can become confusing and difficult to navigate and use. One can be presented with an assortment of devices, without knowing the location of those devices in the physical environment.

AMRE works with the user to filter out the relevant wireless devices from those in the background. Relevance is defined as those wireless devices that are viewable through the EyeTap. This uses the humanistic intelligence to dictate that the interesting objects are the ones in the current field of view. AMRE wirelessly locates such devices and provides labels for them through the EyeTap reality mediator, enabling the user to easily select a specific device out of all the available wireless devices around them. The device, once discovered and identified, is able to be used through a wireless technology. Information such as a computer generated label or graphical user



Figure 4.1: Laptop viewed and identified through IrDA. The text “DebianLaptop” is the computer generated text that is shown in the EyeTap.

interface is made available.

### 4.2.1 Implementation

#### System Overview

AMRE consists of three parts: initial discovery, visual identification, and interaction.

1. Discovery is an ongoing process that recognizes when a device comes into view of a user’s EyeTap. The placement of the IrDA transceiver is used to determine what devices are in EyeTap view and are considered relevant to the user. It was placed directly on the EyeTap in order to achieve this and provides the context of what is currently of interest to the user. The IrDA protocol handles the discovery of other IrDA compliant devices automatically.
2. Visual identification makes use of the information from the discovery process. This information is transmitted from the device and is used to generate a label and provide the necessary information to identify what remote devices are currently available to the user. The wearable computer performs this operation continually and presents the located devices to the user

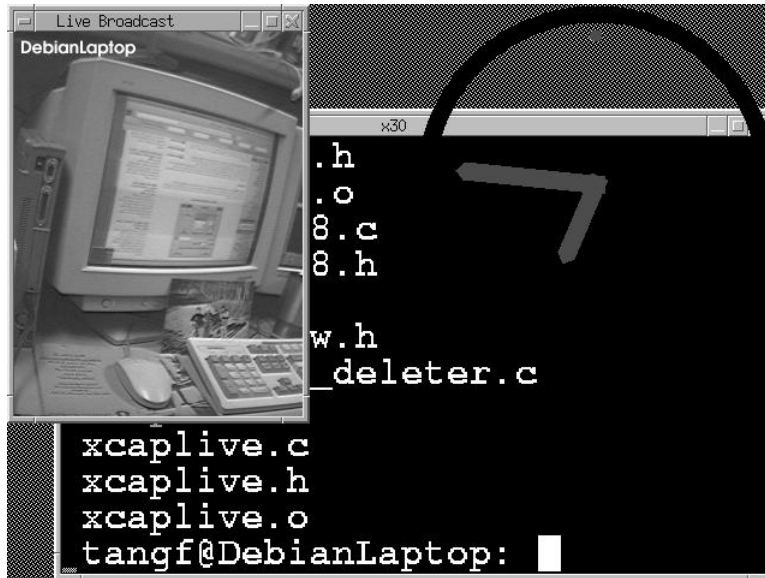


Figure 4.2: The view through the EyeTap composed of an xcaplive display and a remote GUI window connected to the Laptop.

through the EyeTap in the form of text labels in the display as requested. At this point the system has enough information to begin interacting with the device as illustrated in Figures 4.1 and 4.2.

3. Interaction with the device is controlled by the EyeTap user through the wearable computer and all information presented is transferred from the remote device as illustrated in Figure 4.2. This information is transmitted from the device of interest and does not need to be stored in the wearable computer or retrieved through an external network. It can use the IrDA link or a high speed non line of sight wireless communications hardware such as Bluetooth or 802.11b.

With the use of Bluetooth or 802.11b, interaction with the device can continue through a high speed link that is non line of sight. This allows for interaction from a considerable distance away. This also allows for the use of the device even when one is not able to “see” it, this is desirable in familiar environments such as the home or office; a user will know what device names correspond to which devices. Through the use of the wearable computer, these names can be made defaults and if not available other devices can be searched out.

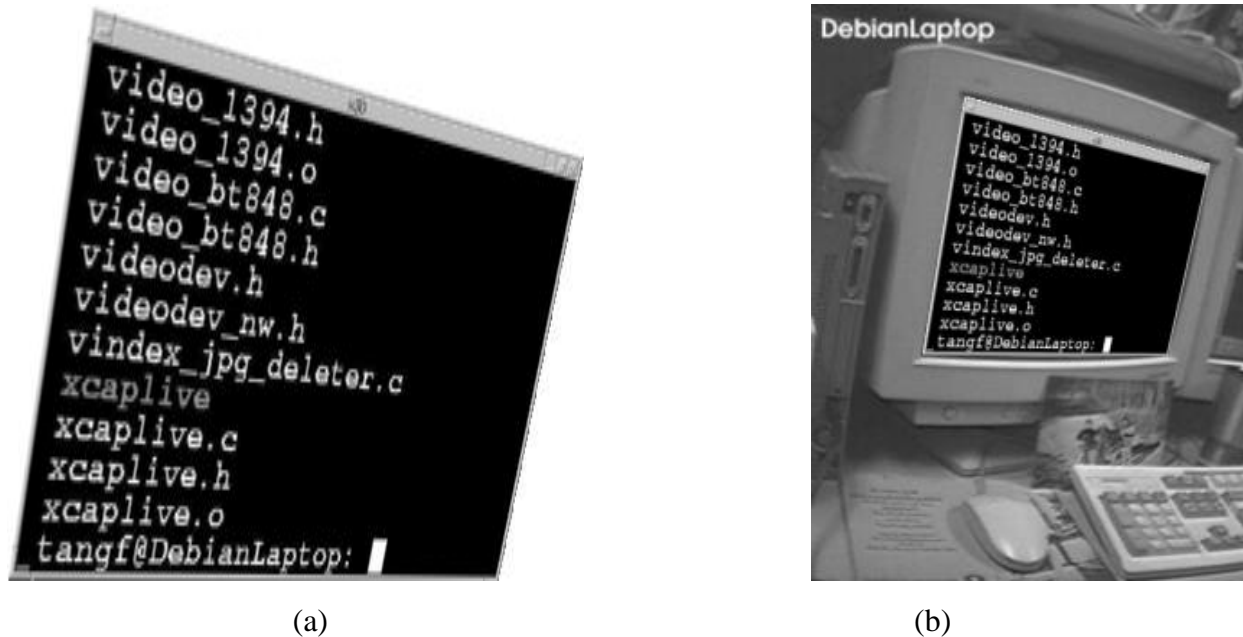


Figure 4.3: (a) The projectively transformed image that is used to mediate the scene. (b) A reality user interface (RUI): the ideal way to incorporate the GUI into the existing environment for a true mediated reality.

### 4.3 Mediated Reality

Ideally, to have an operationally constant interaction with the world the GUI window would be placed into the image stream as illustrated in Figure 4.3. The result is a reality user interface [1]. In this case the world is diminished because there is the removal of information and at the same time it is augmented through the addition of computer generated information, in effect creating a mediated reality [16]. The difficulty lies in the placement of the window as this information must be known a-priori.

### 4.4 Summary

The AMRE system allows for the presentation of textual labels or graphical user interfaces from wireless devices that are in the user's view. This is accomplished through the use of an EyeTap reality mediator and the use of humanistic intelligence.



# Chapter 5

## Creating Mediated Reality with Hardware Accelerated Image Registration

Many current graphics chipsets incorporate hardware specifically designed to achieve fast real-time rendering of texture mapped polygons as well as hardware designed for filtering and pixel interpolation to create accurate texture maps. In particular, graphics chipsets are tuned to create perspective projections of planar surfaces. Therefore by texture mapping the image to a planar surface and then applying the desired projection, the image will be projected by the computer graphics hardware. This will be shown to be faster than doing the projections in software.

In this way, the computer graphics hardware, which is usually used for image synthesis, is instead being used for the purpose of accelerating a computer vision algorithm (image analysis) and in the construction of a shared mediated reality.<sup>1</sup>

### 5.1 Background

#### 5.1.1 VideoOrbits and Projective Flow

Motion is information that is available in a time varying sequence of images. This motion can be caused by movement in the scene, in the case of a static camera, or by the movement of the camera itself (i.e. egomotion).

---

<sup>1</sup>This work has also been presented in [17].

It is understood that the image of the scene is a projection of the 3D world onto a 2D surface. As the object traverses through the scene it will project onto different parts of the image. Thus the vector that connects these 2 locations on the image in relation to the vector that represents the motion of the actual object can be defined as the *motion field*.

“Optical flow is the distribution of apparent velocities of movement of brightness patterns in an image” [18]. Optical flow is not the same as the motion field. For example, when a white disc is spinning on a white background, the optical flow is zero but motion field is not. In most cases the optical flow is equivalent to the motion field. We make the assumption that the optical flow that can be calculated will be representative of the motion field. In fact, we will be assuming a gradient (*i.e.* image brightness) based approach to calculating optical flow. The gradient based method uses spatial-temporal partial derivatives to estimate the optical flow.

Assuming, that the brightness of the image does not change,  $E(\mathbf{x}, t)$ , this would also require that shadows and illuminations are not changing and that the object does not deform greatly as it travels through the scene:

$$E(\mathbf{x}, t) = E(\mathbf{x} + \Delta\mathbf{x}, t + \Delta) \quad (5.1)$$

Therefore each point in frame  $t$  is a translated version of the point in frame  $t + \Delta t$ , and  $\Delta\mathbf{x}$ , and  $\Delta t$  are in the ratio  $\mathbf{u}_f = \Delta\mathbf{x}/\Delta t$  and this is the translational flow velocity at that point. In the case of a single motion of the camera and no independently moving objects in the scene,  $\mathbf{u}_f$  is constant across the scene. Expanding equation 5.1 in a Taylor series and dropping higher order terms, we get the brightness constancy constraint equation (BCCE):

$$\mathbf{u}_f^T \mathbf{E}_x + E_t \approx 0 \quad (5.2)$$

where  $E_x = dE(\mathbf{x}, t)/dx$  is the spatial derivative and  $E_t = dE(\mathbf{x}, t)/dt$  is the temporal derivative. VideoOrbits is a direct featureless method for estimating the “exact” projective coordinate transform that registers pairs of images [19]. A projective coordinate transform exactly describes the motion in two cases of static scenes: (1) images taken from the same location of a 3D scene with a camera that is free to pan, tilt or rotate about its optical axis and (2) images of a flat scene taken from arbitrary locations.

A common assumption in optical flow for computer vision is that the coordinate transform between frames is affine. Although an affine model will account for rotation about the optical axis

of the camera, zoom of the camera, and pure shear, the affine model cannot capture pan and tilt and cannot therefore show the “keystoning” and “chirping”. However, the 8 parameter projective model gives the desired 8 parameters that exactly account for all possible zero-parallax camera motions including keystoning and chirping.

When the change from one image to another is small, optical flow may be used. When minimizing the error of the Horn and Schunk brightness constancy constraint equation the model for the flow is projective. Where the projective coordinate transform (PCT) is defined as,

$$\mathbf{x}' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \frac{\mathbf{A}[x, y]^T + \mathbf{b}}{\mathbf{c}^T[x, y]^T + 1} = \frac{\mathbf{A}\mathbf{x} + \mathbf{b}}{\mathbf{c}^T\mathbf{x} + 1}. \quad (5.3)$$

This is solved by substituting,

$$u_m = x' - x = \frac{ax + b}{cx + 1} - x \quad (5.4)$$

into equation 5.5. However, there will be a discrepancy between the projective model velocity  $u_m$  and the actual flow velocity  $u_f$  due to the violations of the optical flow constraints (e.g. parallax, shadow movement, noise from CCD sensors). Therefore we apply the technique of linear regression and solve for a best least-squares fit by minimizing,

$$\varepsilon = \sum \left( \left( \frac{\mathbf{A}\mathbf{x} + \mathbf{b}}{\mathbf{c}^T\mathbf{x} + 1} - \mathbf{x} \right)^T \mathbf{E}_x + E_t \right)^2 \quad (5.5)$$

where  $E_x$  and  $E_t$  are the spatial and temporal derivatives.

Minimizing (5.5) is simplified by weighting by  $(\mathbf{c}\mathbf{x} + 1)$ , giving:

$$\varepsilon_w = \sum \left( (\mathbf{A}\mathbf{x} + \mathbf{b} - (\mathbf{c}^T\mathbf{x} + 1)\mathbf{x})^T \mathbf{E}_x + (\mathbf{c}^T\mathbf{x} + 1)E_t \right)^2 \quad (5.6)$$

where  $\varepsilon_w$  denotes the weighted error.

To solve for the minimum, we differentiate with respect to the free parameters  $\mathbf{A}$ ,  $\mathbf{b}$ , and  $\mathbf{c}$ , and set the result to zero to give a linear solution:

$$\left( \sum \phi\phi^T \right) [a_{11}, a_{12}, b_1, a_{21}, a_{22}, b_2, c_1, c_2]^T \quad (5.7)$$

where  $\phi^T = [E_x(x, y, 1), E_y(x, y, 1), xE_t - x^2E_x - xyE_y, yE_t - xyE_x - y^2E_y]$ .

This following scheme is often used in with multi-scale Gaussian pyramid in order to account for large motions. The following steps will be run at each level several times and the resulting projective parameters will be passed forward to the next level as the initial estimation.

- (1) **Estimate** the eight parameters of the projective coordinate transform that relates the two image frames,  $\mathbf{p}_k$ .
- (2) **Apply** the projective coordinate transform  $\mathbf{p}_k$  to the appropriate image.
- (3) **Estimate** the projective coordinate transform between the transformed imaged and the other untransformed image,  $\mathbf{p}_{k+1}$ .
- (4) **Resample** use the law of composition to accumulate the effect of the  $\mathbf{p}_k$ 's.
- (5) **Repeat** steps 1-4 until desired accuracy is achieved or maximum number of repetitions is reached.

### 5.1.2 VideoOrbits: Image Registration and Image Composites

An example of VideoOrbits image registration is illustrated in Figure 5.1. The images in Figure 5.1 (a), (c) and (e) are the individual frames of video captured by an EyeTap wearer. The images in Figure 5.1 (b), (d) and (f) undergo a projective coordinate transform to make them appear as viewed from the perspective of the Figure 5.1 (a). The images are registered and composited in Figure 5.2.

## 5.2 Optimizing VideoOrbits

VideoOrbits is well suited for applying OpenGL video acceleration because VideoOrbits is a repetitive multiscale algorithm. Figure 5.3 shows the VideoOrbits algorithm. The steps of the algorithm which can be accelerated with graphics hardware are outlined in bold. For a fully interactive mediated reality system the VideoOrbits algorithm needs to run real-time or at 30 frames per second.

Since, the VideoOrbits algorithm and its implementation is the basis of the VideoOrbits Head Tracker (VOHT), and VideoOrbits Gyroscopic Head Tracker (VOGHT), the optimization of the VideoOrbits algorithm will result in an optimization in both the VOHT and VOGHT and the shared mediated reality system (refer to Chapter 6).



Figure 5.1: (a)(c)(e): Frames of video from an EyeTap device. (b)(d)(f): Images that have undergone a projective coordinate transform to match the perspective of (a).

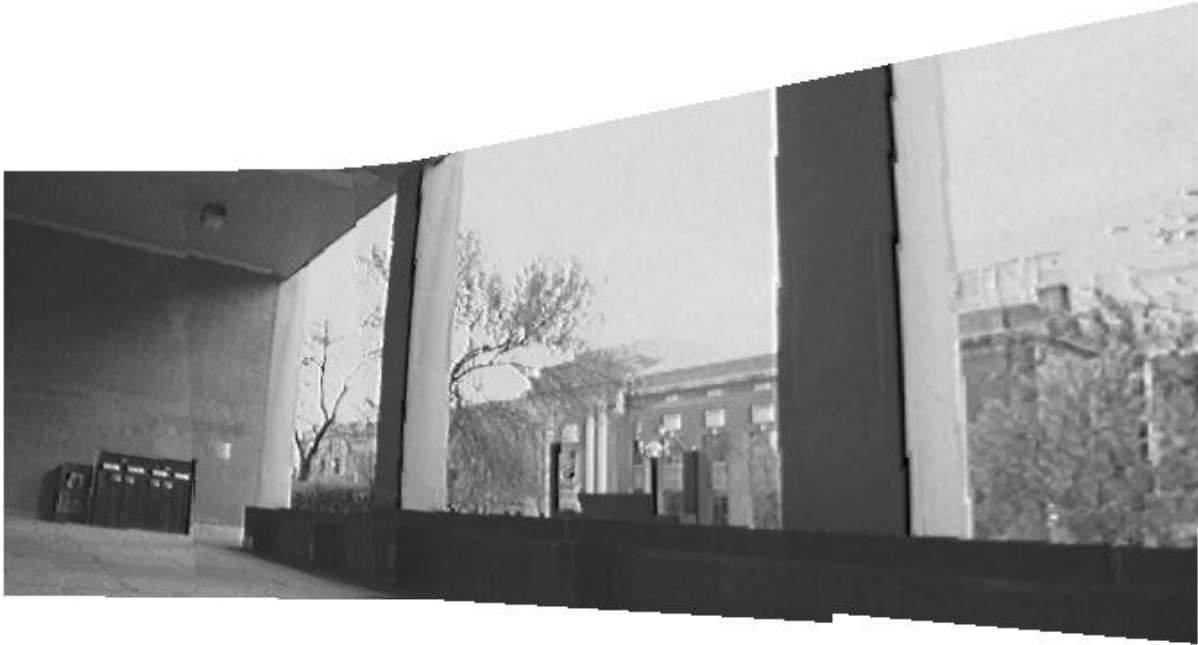


Figure 5.2: The projectively transformed images are registered and “cemented” together into an image composite.

### 5.3 Mapping Projective Coordinate Transformations

In OpenGL, the most straightforward way of applying the projective coordinate transformation of VideoOrbits is to consider equation 5.11 to be a transformation to be applied to the projection matrix used in OpenGL.

The operation of applying a projective coordinate transformation to an image is isomorphic (isomorphic refers precisely to algebraic isomorphisms, as discussed in [20]) to the process of projecting a texture mapped polygon under perspective projection in OpenGL [1]. Thus, hardware acceleration of VideoOrbits projective transformations can be achieved by defining an isomorphism between the projective space of VideoOrbits and the projective space and homogeneous coordinate system of OpenGL. An isomorphism  $\phi$  is defined by a mapping of VideoOrbits projective transformations  $G$  to OpenGL projection matrices  $M$ :

$$\phi : G \rightarrow M \quad (5.8)$$

In the VideoOrbits algorithm, the projective coordinate transformation (PCT) is written as equation 5.3. Thus, it defines an eight parameter space. The transformation can be re-written as a  $R_{3 \times 3}$

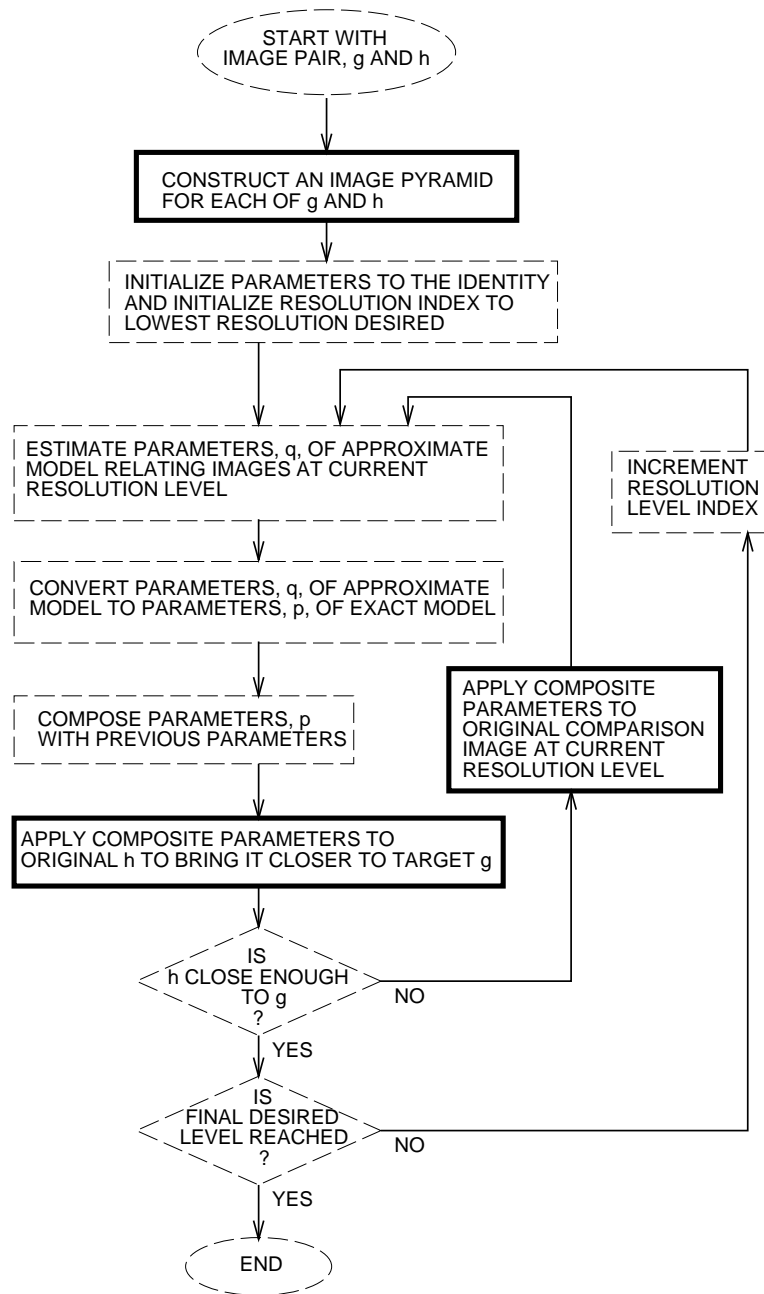


Figure 5.3: The VideoOrbits algorithm. The steps shown in solid dark lines have been accelerated using graphics hardware available on common 3D accelerated computer graphics chipsets. This figure is from [17].

matrix:

$$\begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ c_1 & c_2 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = G\mathbf{x} \quad (5.9)$$

where it can be seen that the set of projective coordinate transformation forms a group acting upon a set  $S$  of image coordinates.

Thus, what is desired is some isomorphism  $\phi$  mapping the projective coordinate transformation of VideoOrbits to a  $R_{4 \times 4}$  projection matrix in OpenGL.

The desired isomorphism is given by:

$$\phi(G) = \phi \left( \begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ c_1 & c_2 & 1 \end{bmatrix} \right) \quad (5.10)$$

$$= \begin{bmatrix} 1/a_{22} & -a_{21} & b_2 & 0 \\ -a_{12} & 1/a_{11} & b_1 & 0 \\ c_2 & c_1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5.11)$$

This mapping takes into account the different coordinate systems and conventions used by each program. Equation 5.11 is used as a camera transformation matrix. Thus, it describes the transformation the camera undergoes, such that the plane will appear as required under OpenGL perspective projection.

To perform the image projection in OpenGL, the image is first loaded into the OpenGL program as a texture map. Then the camera is positioned and the perspective projection is applied. The resulting image is read out of the buffer and the image is stored.

## 5.4 Measurements of OpenGL acceleration

To determine the speed-up attained by using hardware acceleration, a program using the hardware acceleration was compared with the software algorithm. A set of projective coordinate transformations was generated according to the equations of [21]. All programs were run on a wearable computer with a 700 MHz Pentium-III processor, with 64 MB of RAM. The wearable computer had an Intel i810 graphics chipset. This computer was using the GNU/Linux operating system

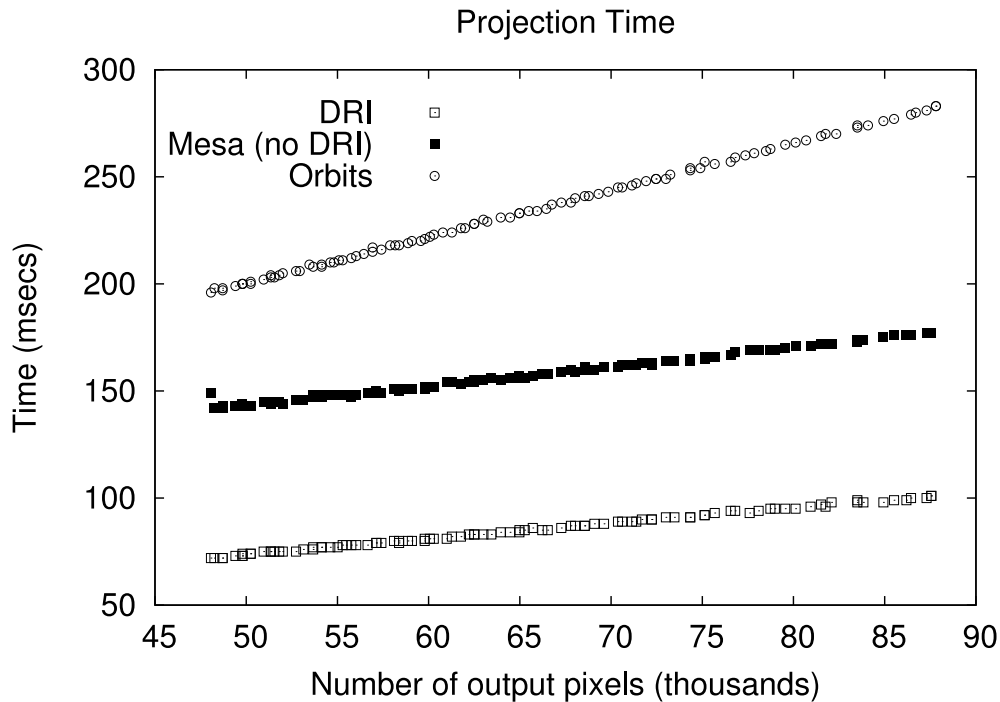


Figure 5.4: Projection times for VideoOrbits software, Mesa3D (using software rendering) and Mesa3D with DRI hardware rendering enabled. This figure is from [17].

and the direct rendering interface (DRI) which allows direct access to the graphics hardware for programs.

Figure 5.4 shows the results of timing a single projection using three methods: 1) a program using Mesa3D and available computer graphics hardware and DRI; 2) a program using Mesa3D, using software algorithms; 3) a program running the equivalent VideoOrbits algorithm.

Thus, an additional program is discussed here, which uses the software implementation of Mesa3D (actually the Mesa3D program is the same as the DRI program, with the direct rendering turned off). The software Mesa3D was examined because it is considered to be well optimized code for computer graphics applications. Thus, computer vision algorithms can also benefit from the speed and optimizations used in computer graphics software. So, on machines which may not benefit from 3D graphics acceleration, Mesa will still implement an optimized software projection, and additionally this was examined.

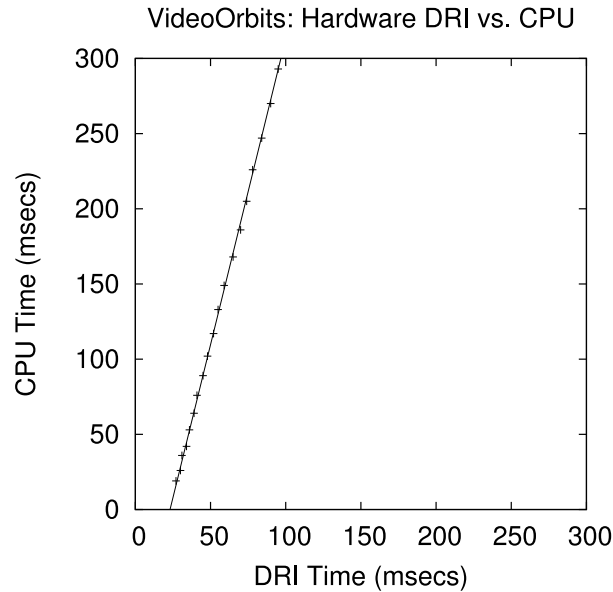


Figure 5.5: Projection times for VideoOrbits using DRI vs. the CPU given varying image sizes. All images had identical projection parameters. This figure is from [17].

For the plot of Figure 5.4, the input image size was set, and different projection parameters were given to the three different programs, and the time taken to project the image was recorded. The projections used were independent rotations about each of the principle axis, with a maximum rotation of 15 degrees about any axis. From the data, the average speedup between VideoOrbits using DRI vs. using the CPU was  $2.75\times$ . The average speedup between VideoOrbits and the Mesa software rendering was  $1.48\times$  and the average speedup between the Mesa software rendering and the DRI implementation was  $1.83\times$ . This speedup verifies that our DRI implementation did indeed use the available graphics hardware.

Figure 5.5 shows the effect of hardware acceleration on input images of different sizes. For this figure, the projection parameters were held constant, and the input image size was varied. In all cases, the hardware accelerated program projected the image faster than VideoOrbits. The smallest image size was  $76 \times 58$  and the largest input image size was  $435 \times 331$ . The slope of a linear best fit line through the plot is 2.99. Thus, for this range of input image sizes, the hardware speedup was  $2.99\times$ .

Figure 5.6 shows the effect of different projection parameters on the speed of the projection.

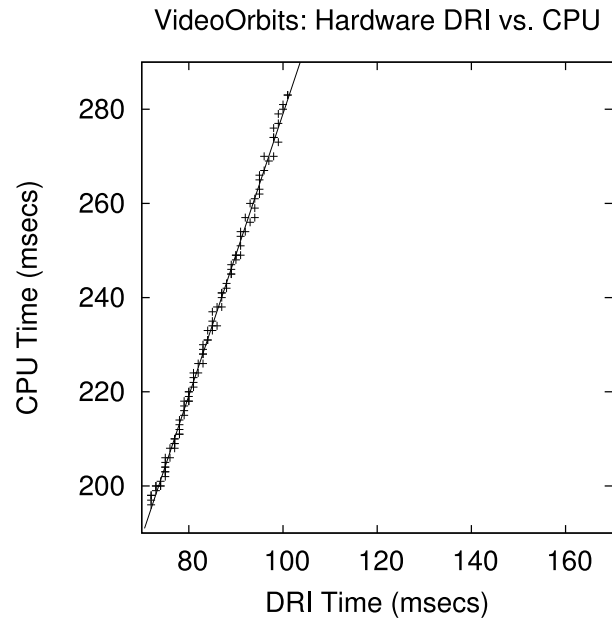


Figure 5.6: Projection times for VideoOrbits programs, one using DRI and the other the CPU. The algorithm was given a fixed input image and the projection parameters were varied (resulting in larger output images). This figure is from [17].

For this plot, the input image was held constant, but the projection parameters were varied. The larger projection times shown correspond with increasing numbers of output pixels of the resulting image. Thus, this plot is measuring the effects of increased amounts of pixel interpolation, since large output images required more pixel interpolation since there were more output pixels. The slope of this graph was 4.07. Slope here may be interpreted as how well each of the programs using DRI and the CPU dealt with more interpolation being required. Thus, the hardware was able to handle increased amounts of interpolation  $4.07\times$  faster than the VideoOrbits software.

## 5.5 Summary

It was demonstrated that current graphics hardware, which is designed to achieve fast real-time rendering of texture mapped polygons, can be used to accelerate the VideoOrbits algorithm. This was done by mapping the projective coordinate transformation into an equivalent perspective projective camera view. This caused the image transformation to occur in hardware rather than in software causing a  $2.75x$  speed-up. This is a first step in the use of computer graphics hardware to allow for real-time (30 frames per second) execution of the VideoOrbits algorithm on wearable computers.

## Chapter 6

# Seeing Eye to Eye: a shared mediated reality using EyeTap devices and the VideoOrbits Gyroscopic Head Tracker

This chapter presents a system which allows wearable computer users to share their views of their current environments with each other. The system uses an EyeTap: a device which allows the eye of the wearer to function both as a camera and a display. A wearer, by looking around his/her environment, “paints” or “builds” an environment map composed of images from the EyeTap device, along with head-tracking information recording the orientation of each image. The head-tracking algorithm uses a featureless image motion estimation algorithm coupled with a head mounted gyroscope. The environment map is then transmitted to another user, who, through his/her own head-tracking EyeTap system, browses the first user’s environment solely by head motion, seeing the environment as though it were their own. As a result of browsing the transmitted environment map, the viewer builds and extends his/her own environment map, and thus this is a data-producing head-tracking system. These environment maps can then be shared reciprocally between wearers.<sup>1</sup>

---

<sup>1</sup>This work has also been presented in [22].

## 6.1 Background

### 6.1.1 VOHT: Video Orbits Head Tracking

Headtracking is important in developing a useful vision driven interface. [23] surveys many types of tracking systems for use in virtual and augmented reality systems. Many of the systems depends on the existence of external beacons attached to the user, or the existence of special sensors in the local environment. This limits the mobility of the head tracking system because the head motion is confined to the area in which the sensors are setup. An alternate approach is the use of computer vision techniques for the recovery of egomotion [24].

Therefore, a VideoOrbits head-tracker [16] is based on the recovery of egomotion, which is the motion of the camera or head in the case of an EyeTap device. The VideoOrbits algorithm performs head-tracking visually, based on a natural environment, and works without the need for object recognition or external beacons. Instead it is based on algebraic projective geometry, and a featureless means of estimating the change in spatial coordinates arising from movement of the wearer's head, as illustrated in Figure 6.3.

Headtrackers have a problem with drift [25]. Drift is when the head tracker measurement gradually shifts. This is especially apparent when the head is kept still and is due to the accumulation of small errors. The projective coordinate transform estimates that are made relative from frame to frame ( $P_n$ ) have errors due to violations of static scene, independently moving objects, non rotational head motion, or the inability to correctly register frames of video.

$$P_{composed} = \prod_{n=1}^N P_n \quad (6.1)$$

where  $P_{composed}$  and  $P_n$  are expressed in matrix form:

$$P = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{c}^T & 1 \end{bmatrix} \quad (6.2)$$

So,  $P_{composed}$  accumulates the error present in each  $P_n$  and is illustrated in Figure 6.1.

The main issue is that head tracking, by comparing images in a pairwise fashion, is running an open loop system with no feedback mechanisms. A solution is the implementation of the "reference frame" [27, 28] as illustrated in Figure 6.2. The frames of video, instead of being compared pairwise are now each compared against a reference frame. The reference frame is

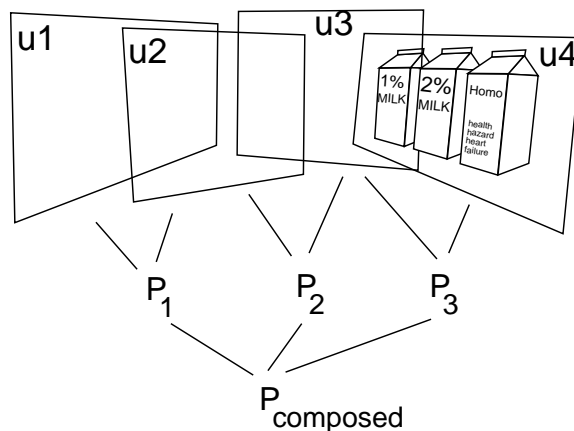


Figure 6.1:  $P_{composed}$  is composed of projective coordinate transforms  $P_1$ ,  $P_2$  and  $P_3$ .  $P_{composed}$  will transform image  $U_4$  to the perspective of  $U_1$ .

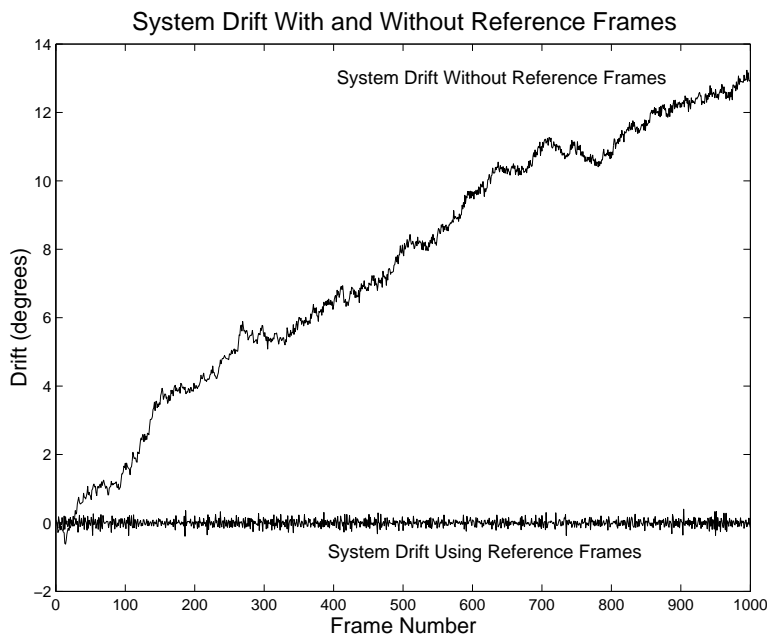


Figure 6.2: This graph shows the effect of reference frame use on system drift. For this experiment, the camera was held in a fixed position. This figure is from [22] and [26].

chosen to be the first frame initially and can be updated or changed when certain conditions are met (e.g. a displacement generates an error which exceeds a threshold, which will set a new reference frame). The reference frame also allows for some robustness in the case of an independently

moving object. This effectively creates a closed loop system in which small errors are prevented from accumulating.

The VOHT is able to calculate the egomotion of the head to sub-pixel accuracy [28] when the magnitude of the motion is small. However, when the motion is large, the optical flow assumptions will break down and image registration will not occur and therefore the egomotion is unrecoverable.

### 6.1.2 VOGHT: Video Orbits Gyroscopic Head Tracker

The VideoOrbits algorithm is able to calculate the egomotion of the camera based on algebraic projective geometry and projective flow [16]. Projective flow is most accurate when the motion between images is small. The head motion of a typical user can vary between large motions (e.g. sweeping the head from side to side) and small motions (e.g. even looking intently at a single object will result in some motion of the head). If the motion is large enough so that there is insufficient overlap between two adjacent frames of images, then the VideoOrbits algorithm will be unable to calculate the projective coordinate transform that relates the two images, and recovery of egomotion is impossible.

The use of a pair of small, low cost, vibrating element gyroscopes with the VOHT allows for the estimation of these relatively large motions of the head [26]. The rotational motion of the head as measured by the gyroscopes can be used as an initial estimate for the VideoOrbits algorithm. The motion estimated by the gyroscope is converted into a projective coordinate transform through:

$$\mathbf{WRW}^{-1} = \lambda \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{c}^T & 1 \end{bmatrix} = \lambda P_g \quad (6.3)$$

$$\text{where } \mathbf{W} = \begin{bmatrix} \frac{f_x}{S_x} & 0 & \frac{O_x}{S_x} \\ 0 & \frac{f_y}{S_y} & \frac{O_y}{S_y} \\ 0 & 0 & 1 \end{bmatrix}, \quad (6.4)$$

$\mathbf{R}$  is the rotation matrix in Euler angles corresponding to the gyroscope rotations in the camera coordinate system.  $f_x$  and  $f_y$  are the camera focal lengths,  $S_x$  and  $S_y$  are the dimensions of the video images,  $(O_x, O_y)$  is the point of intersection of the optical axis of the camera and the sensor array (in units of pixels), and  $\lambda$  is a scale factor used to normalize the last entry in the projective

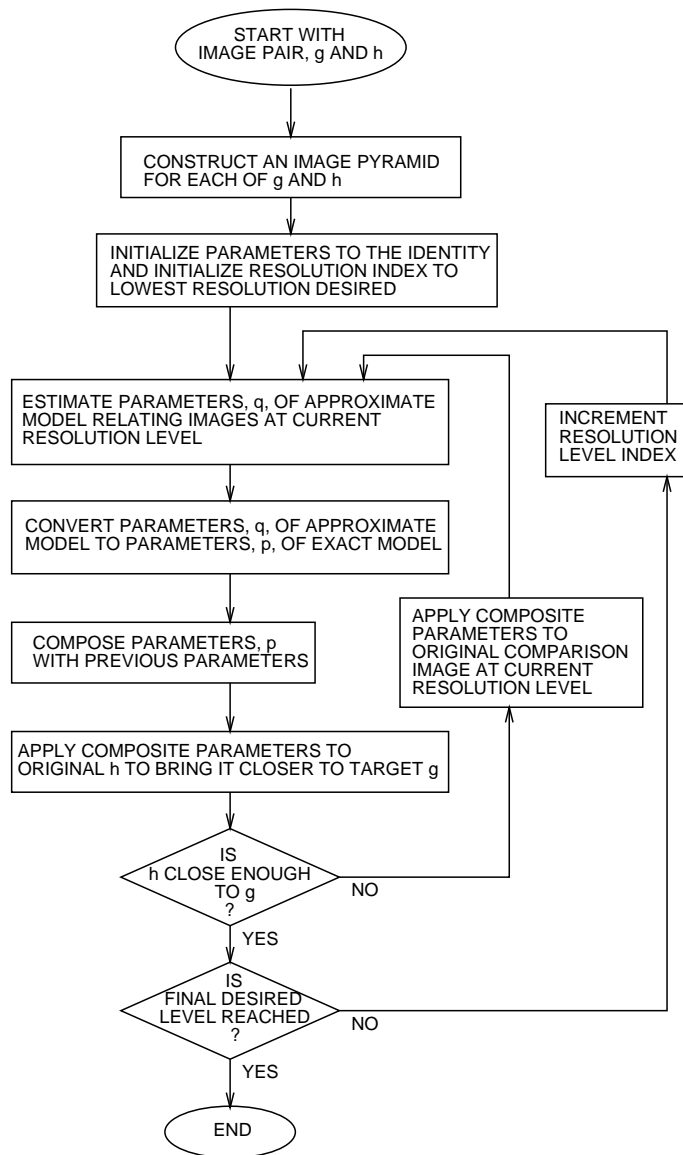


Figure 6.3: **The ‘VideoOrbits’ head-tracking algorithm:** The new head-tracking algorithm requires no special devices installed in the environment. The camera in the Personal Imaging system simply tracks itself based on its view of objects in the environment. The algorithm is based on algebraic projective geometry, and provides an estimate of the true projective coordinate transformation, which, for successive image pairs is composed using the projective group [19]. Successive pairs of images may be estimated in the neighbourhood of the identity coordinate transformation of the group, while absolute head tracking is done using the exact group by relating the approximate parameters  $q$  to the exact parameters  $p$  in the innermost loop of the process. This figure is from [16].

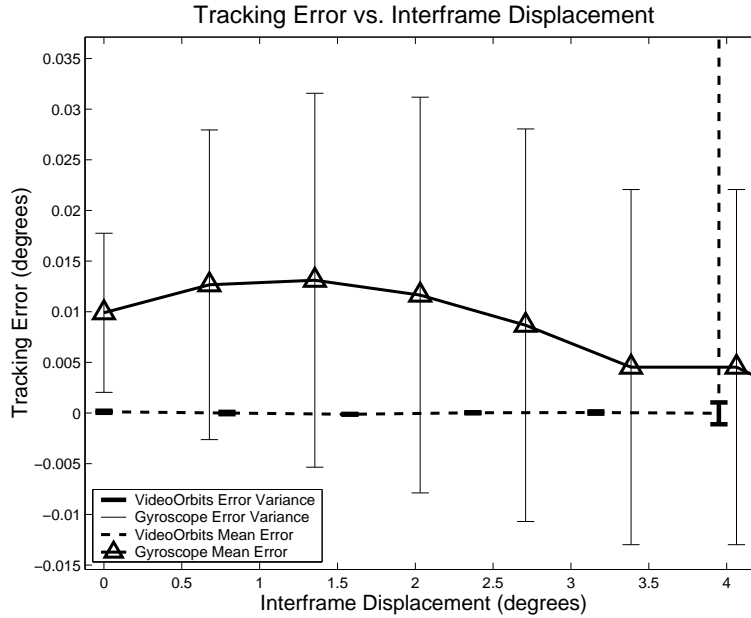


Figure 6.4: This graph compares the tracking ability of VideoOrbits and the gyroscope used. The error bars represent the variance in the tracking error and are displayed as one tenth the actual size. For this experiment, the execution time for VideoOrbits was limited to 0.3 s per frame. Under these circumstances, we can see that if the interframe displacement is less than four degrees, the algorithm converges, resulting in very small error. Beyond four degrees, convergence is not achieved, resulting in unusable PCT estimates. While not shown, the error in the gyroscope remains fairly constant beyond four degrees of interframe displacement. This figure is from [22] and [26].

coordinate transform matrix to one.  $\mathbf{A}$ ,  $\mathbf{b}$  and  $\mathbf{c}$  are the projective coordinate transform parameters defined in Equation 5.3.

The VideoOrbits Gyroscopic Head Tracker (VOGHT) calculates the absolute camera motion by calculating the PCT that relates the current frame to a base frame as done in the VOHT.

The gyroscope is able to convert a large rotational (i.e. head) motion into a PCT which is used as an initial guess for the VOHT. This is done by composing a new PCT estimate ( $P_e$ ) which is to be applied to the new frame of video before estimating its PCT with respect to the reference frame.

$$P_e = \lambda P_r^{-1} P_l P_g, \quad (6.5)$$

where  $P_g$  is the PCT estimated by the gyroscope,  $P_l$  is the PCT describing the position of the

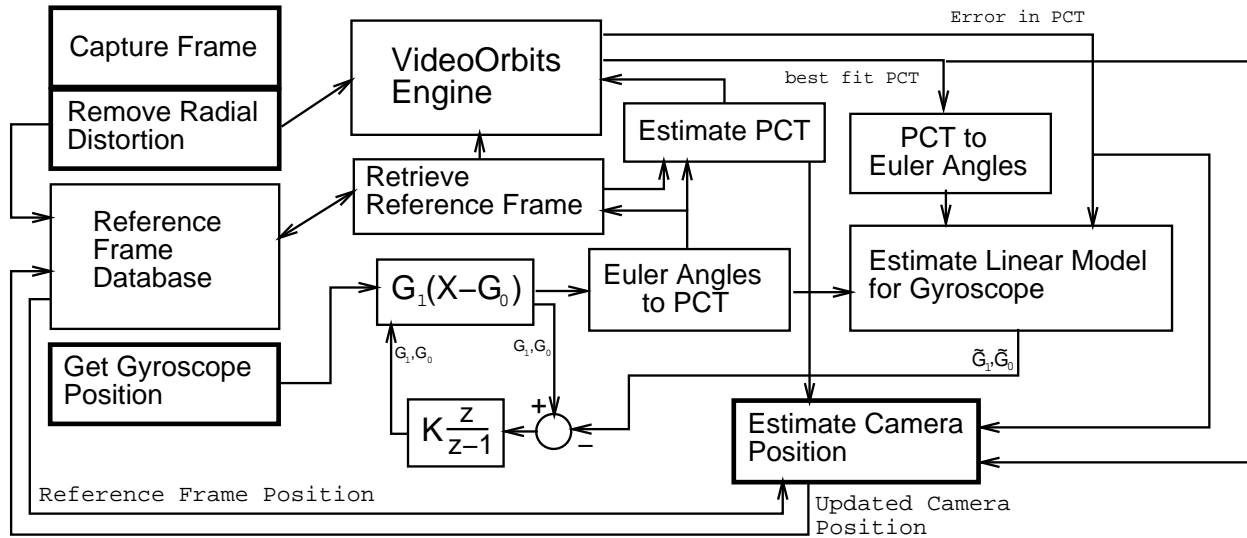


Figure 6.5: Gyro / VideoOrbits Tracker Block Diagram. This Figure is from [22] and [26].

last frame of video and  $P_r$  is the PCT describing the position of the reference frame and  $\lambda$  is a scale factor to normalize the last entry of the PCT to one.

The gyroscopes in the VOGHT system are constantly drifting (i.e. running open loop) and need to be corrected for. However, the drift is small and can be accounted for through the use of the VideoOrbits algorithm. This is done by comparing the PCT calculated by VideoOrbits with that supplied by the gyroscope. If the motion of both is small then the VideoOrbits estimate is used. When the motion is large, the mean square error returned by VideoOrbits is examined, if the error exceeds a threshold then poor registration is assumed and the gyroscope PCT estimate is used.

The gyroscope is computationally simple and is able to provide useful information on a range of rotational head motions. Unfortunately, it is inaccurate and is subject to constant and fluctuating drift. When combined with the VOHT, the gyroscope and VideoOrbits algorithm complement each other as illustrated in Figure 6.4. The VideoOrbits algorithm is able to accurately calculate the PCT for small motions and the gyroscope can provide a useful estimate of the PCT in large motions. The estimate is used as an initial guess for the VideoOrbits algorithm, or it can be used to approximate a PCT in the case of pure rotational motion.

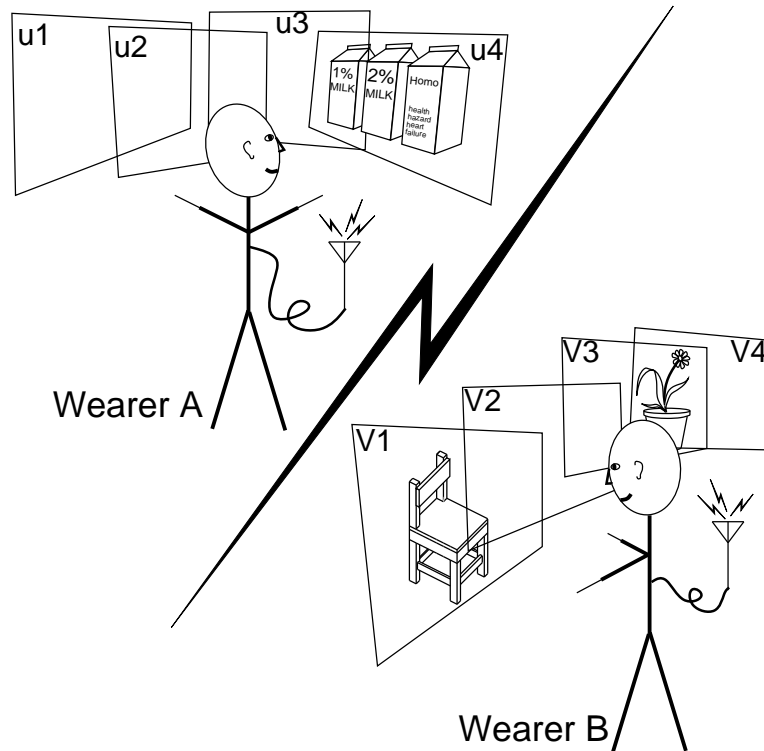


Figure 6.6: EyeTap wearer A, EyeTap wearer B. This figure is from [22].

## 6.2 Shared Mediated Reality

EyeTap devices have been used for the transmission of live video captured from the user's perspective of the surrounding environment. In a previous study, EyeTap reporters accomplish wireless Electronic News Gathering[29] that allow viewers to remotely experience events as if they were actually present.

In contrast, the system that we now propose, similarly allows users to see through each other's eyes, yet without the constraint of following exactly the broadcaster's point of view. This development is extremely important, since exactly following another gaze of shaky EyeTap video can be disorienting and nauseating. The way we scan our environment is very strongly tied to the physical motion that we are currently experiencing, together with the vestibular cues. Observation of live eye or head mounted video is generally much less stable when viewed vicariously than what we seem to experience in reality since our perceptual system does much of the stabilization based on our physical motions, and vestibular cues [1].

The shared mediated reality system allows users wearing EyeTap devices to share their visual experience. As each user looks around, they create a visual/temporal map of their surrounding environment. By sharing this visual history, users can see through the eyes of another. They are able to look about without being constrained to a single point of view, rather being able to generate new projective views from the visual/temporal environment maps. In Figure 6.6, Wearer A, in a grocery store, is building an environment map  $(u_1, u_2, u_3, u_4)$  that includes an image of milk cartons. Wearer B is browsing wearer A's environment map at home. The images  $(u_1, u_2, u_3, u_4)$  are used to generate perspectively correct images that correspond to the views of  $(v_1, v_2, v_3, v_4)$  and are displayed in the positions  $(v_1, v_2, v_3, v_4)$ , replacing wearer B's view of the chair and plant. At the same time, wearer B is incidentally constructing an environment map of the home, which would allow wearer A to view the chair and plant from Wearer B's position.

People often scan their environment with quick head rotations. As a result, a large percentage of EyeTap video can be described (with a high degree of accuracy) by a camera moving about a fixed center of projection in a static scene. Using this simplification and applying the VideoOrbits algorithm to projectively transform the images, images from arbitrary camera orientations can be synthesized by forming a composite of spatially relevant neighbouring images. The vicariously experienced environment can thus be observed from any angle, allowing the user to navigate the virtual environment by rotating their head as if they were actually there.

### 6.2.1 VideoOrbits

Since the motion of a head mounted camera or EyeTap between successive frames of video can be approximated very well by a pure rotation, a projective coordinate transformation can be used to register a pair of overlapping images with sub-pixel accuracy [1]. Well registered images can then be combined to create image composites of greater spatial extent. By generating this composite image with appropriately projective coordinate transformed images and cropping the result, arbitrary camera orientations can be simulated as illustrated in Figures 6.7 and 6.8, therefore allowing a user to view another's environment without being limited to previously held viewpoints.

While this technique is exact for situations where each user views their world through pure rotation, performance in the non-ideal case is improved by the extra five degrees of freedom provided by the 8 parameter projective coordinate transform. The extended freedom allows for visually acceptable composites to be formed in many cases where the pure rotation condition has been clearly

violated. This is especially true in outdoor urban environments where many video sequences have dominant planar surfaces.

### 6.2.2 Head Tracking

In the proposed shared mediated reality system, a user may navigate another's environment by rotating their head to achieve a desired point of view. The user's head motions are approximated by pure rotations, since the shared environment map does not easily provide depth information. The images that comprise the environment map are stored with the associated rotational position that defines the orientation in which they were captured. This method of browsing the environment map provides the experience of seeing through the other user's eyes.

The proposed shared reality system requires the projective coordinate transform between images as well as rotational orientation of the camera during their capture. This information is provided by a modified version of the VOGHT.

### 6.2.3 VOGHT: Reference Frame Database

The VOGHT was extended to use a spherical grid of reference frames. This is necessary when the head motion will cause the current frame of video to have no overlap with the current reference frame. At this point the VideoOrbits algorithm will be unable to calculate the PCT that relates the two images.

The Reference Frame Database is a spherically indexed database of reference images of the environment that are continuously collected and updated through normal system use. Each image in the database includes a time stamp, a 3-D rotation matrix ( $R_r$ ) describing the camera orientation, a PCT ( $P_r$ ) describing its projective position with respect to the reference frame. Reference frames are selected for use with VideoOrbits based on the current position estimate generated by the gyroscope.

It is important to note that if large head rotations occur, it is possible for consecutive image frames to have no overlap at all. This can prevent VideoOrbits from estimating a useful PCT in the case where the sphere of environment images is poorly populated and no nearby reference images exist. In this case, a new reference image will be stored with a gyro-generated PCT. New reference images with gyroscope generated PCTs can also be created if VideoOrbits cannot find

good registration with an existing reference frame.

Images whose associated PCTs were generated by VideoOrbits are selected in preference to those whose positions were estimated by the gyroscope alone. More recent images are selected in preference to older images to account for environment changes. Each time a reference frame is used successfully, its timestamp is updated to ensure that it continues to be used instead of its neighbours.

Reference frames are saved when the current video image is offset from the current reference frame by a significant amount. This amount is dependent on the camera's field of view and the system resources of the operating platform. The sphere of reference images is mapped to a two dimensional matrix, indexed by equal increments of azimuth and elevation. In experimentation, two degree increments were used. The remaining entries of the matrix are marked as NULL, as they are redundant. Searching this matrix is simple since the camera position can be directly related the position in this matrix. If the search includes NULL entries, their nearest neighbours (of greater azimuth and elevation) are used. It is important to note that large camera velocity at the time of image capture can generate image blurring which negatively impacts the functioning of the VideoOrbits algorithm. These blurred images are generally rejected due to their high MSE results.

### 6.3 Seeing Eye to Eye: a shared mediated reality

The requirement for the shared mediated reality system to store, retrieve and spatially register images is satisfied by making use of the spherical reference frame database that is maintained by the VOGHT for head tracking purposes. The reference frame images are stored along with the rotational orientation of the camera  $R_r$ , projective parameters  $P_r$ , and indexed by integer values  $(\theta, \phi)$ , representing the azimuth and elevation of the camera orientation.

Sharing of the mediated reality is accomplished by transferring this reference frame database to wearer B, who can use a second VOGHT and a browser program to synthesize views of wearer A's environment map from an arbitrary viewing direction. As long as the environment map contains some reference images in the neighbourhood of the precise direction desired, the browser program will display a correctly projected view.

Synthesis of views is performed by taking the current estimate of the rotational position ( $R$ ) of wearer B's VOGHT and using its equivalent PCT ( $P$ ) to re-project wearer A's reference im-

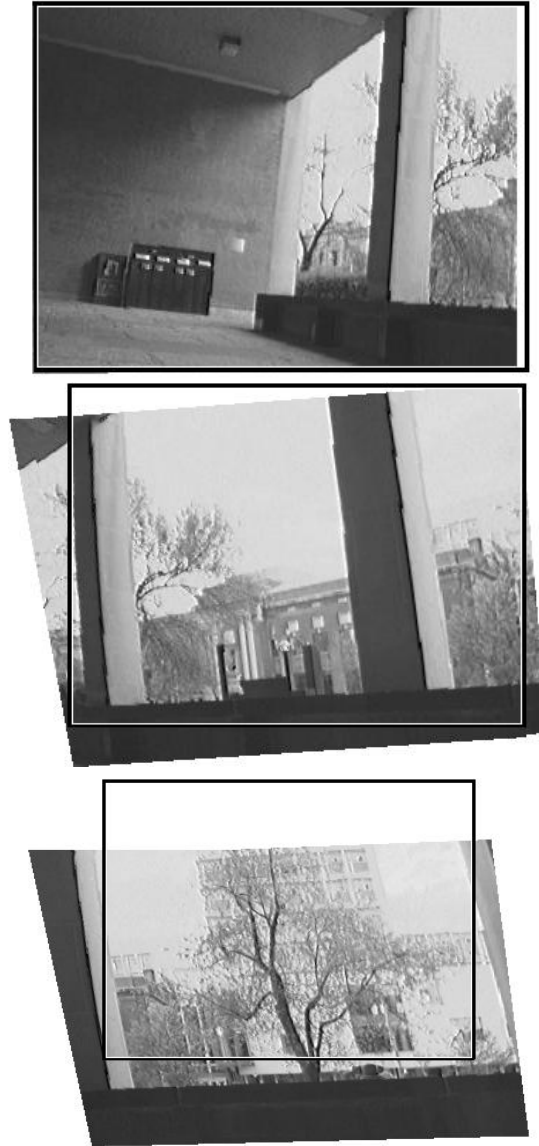


Figure 6.7: Outdoors: three example views by wearer B of the environment map that was synthesized from a single reference frame from wearer A.



Figure 6.8: Outdoors: the environment map generated by wearer A, and the views of that map synthesized by wearer B using multiple reference frames. This figure is from [22].

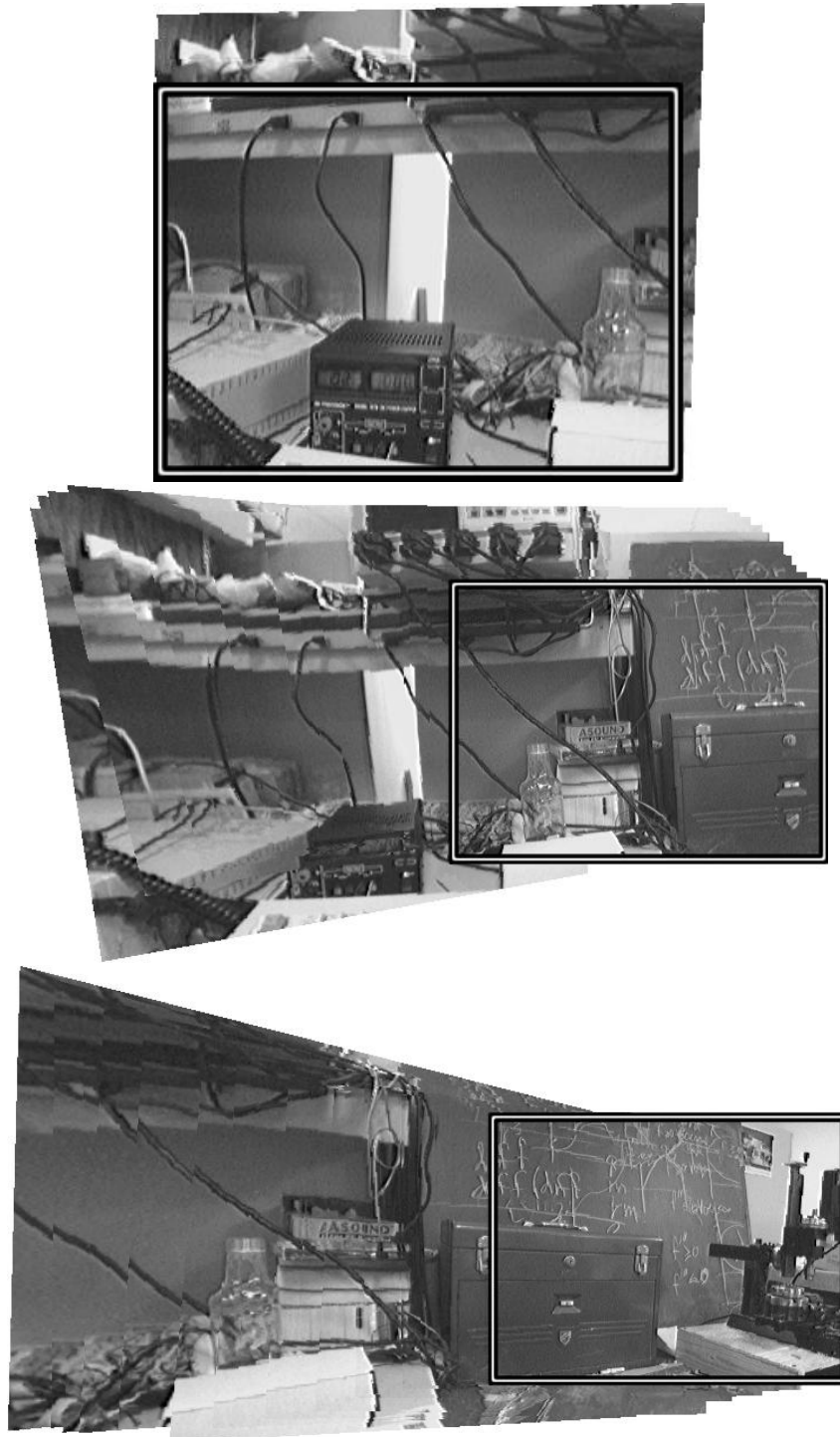


Figure 6.9: Indoors: the environment map that wearer B generates while browsing wearer A's outdoor map. This figure is from [22].

ages to wearer B's viewpoint. In order to increase the creation speed of the synthesized view, a bounding box is computed and only the reference images that will contribute to the final view are re-projected. In order to minimize composition error due to user translation and independently moving objects, the images are composed in chronological order such that the most recent images are placed on top.

Thus the head tracking of wearer B is used to provide a perspective with which to view the environment map generated by wearer A. As wearer B rotates his/her head, new perspectives are supplied to the browser program, which cause the projection and synthesis of new portions of the environment map. All these projections are relative to wearer B's current head position, thus all navigation is performed solely with head movement.

The computation of the bounding box can be adjusted to the available computational resources, balancing final image quality against execution time. On a resource constrained wearable computing system, the projective coordinate transform may be applied only to the single closest and newest image as illustrated in Figure 6.7.

The resulting image will be quite complete if the image from the map is near the desired viewing perspective. If the image from the map is distant from the viewing perspective, then the generated image will not be complete due to the fact that there is no information available.

If the search area is large, then multiple images from the map can be projectively transformed and registered (see figures 6.8 and 6.9). This registered image allows for the creation of a complete image for the desired perspective even if there is no image in the map at that exact perspective.

The images that are projected are selected by the position  $(\theta, \phi)$  of the VOGHT. This estimate is used to calculate a region in which to apply the projective coordinate transform. Then using the projected images an image composite is formed. From this composite the correct view is cropped. A range of images is used in order to use a sufficient amount of data from which to generate the new frame.

The gyro position of the image does not affect the calculation of the projection of the images, it is used only to select the region the scene will be constructed from. The search size of the area is a tunable parameter, a small parameter is useful on a resource constrained wearable computing system. This will apply the projective coordinate transform to the closest and newest image. The result if the image is near the view will be quite good, however if there is no image nearby then no image composite can be generated.

### 6.3.1 Demonstration

The system may be seen in action in figures 6.8 and 6.9, demonstrating results of wearer B browsing an environment map created by wearer A. The first image in figure 6.8 shows the environment map that wearer A has generated out of doors on a veranda. The bounding box indicates the portion that wearer B is currently viewing. The first image of figure 6.9 shows the corresponding view of B's environment inside a laboratory, with the bounding box described by the EyeTap. The middle images show the result of looking to the right: in figure 6.8 the view has moved to the middle of the environment map, while in figure 6.9, we see that wearer B is simultaneously mapping out their own environment. The final pair of images carries this process further, and also reveals how the system handles the case of insufficient reference frames to fill a view completely: there is a narrow band of white at the top of the bounding box in figure 6.8 indicating an area with insufficient information in the environment map.

## 6.4 Accounting for the Camera Intrinsic Parameters

The EyeTap devices are custom made prototypes and as such contain different cameras. In a shared mediated reality, the remote environment maps will be generated by the remote EyeTap with camera intrinsic parameters  $M_{int_{rem}}$ . The browser will have an EyeTap with camera intrinsic parameters  $M_{int_{loc}}$ . Therefore to create the view of the remote environment through the EyeTap of the local user, the intrinsic parameters of both cameras have to be accounted for in equation 6.6. This allows not only the correct projective transformation but accounts for the differences in each EyeTap's camera intrinsic parameters.

$$P_{view} = P_r^{-1} M_{int_{loc}} M_{int_{rem}}^{-1} P_n, \quad (6.6)$$

$$M_{int} = \begin{bmatrix} -f_x & 0 & O_x \\ 0 & -f_y & O_y \\ 0 & 0 & 1 \end{bmatrix}, \quad (6.7)$$

where  $f_x, f_y$  are the focal length and  $O_x, O_y$  is the location of the image center in pixel coordinates.

## 6.5 Summary

The Eye to Eye shared mediated reality system allowed users to exchange their current environments through the EyeTap reality mediator. The views were generated and controlled through the user's head motion using the VOGHT. The VideoOrbits algorithm was integral in the system and was used to calculate the projective coordinate transforms between images as well as in image registration for the creation of image composites.

Due to the constrained computing power of the wearable computer and the limited bandwidth of the wireless connection, it is not feasible to transmit every captured frame. Through the use of a head tracker, the frames captured and transmitted are optimized to span the entire sphere around the user in an efficient manner.

This was demonstrated through the use of the system between two wearable computer users, one outdoors and one indoors. The views generated had the correct perspective and allowed each viewer to see the other's environment as if it were their own. This resulted in the users seeing Eye to Eye in a projectively stabilized shared mediated reality.



# Chapter 7

## Conclusion

Through the construction of an EyeTap device and wearable computer, a mediated reality system was implemented. The understanding, use and integration of wireless technologies such as Bluetooth, 802.11b and IrDA in GNU/Linux was done in order to create a network of wearable computers.

The line of sight IrDA and non line of sight Bluetooth and 802.11b wireless technologies with the use of EyeTap reality mediators allowed for the interaction between the wearable computer user and objects in the environment. The objects mediated the reality of the EyeTap user at his/her discretion by providing information such as labels and graphical user interfaces.

Interactions between wearers of EyeTap devices was explored in Chapter 6. This allowed the user, by looking around his/her environment, to “paint” or ”build” an environment map composed of images from the EyeTap device, along with head-tracking information recording the orientation of each image. The environment map is then transmitted to another user, who, through their own headtracking EyeTap system, browses the first user’s environment solely by head motion, seeing the environment as though it were their own.

The VideoOrbits algorithm formed the basis of the *Seeing Eye to Eye: a shared mediated reality using EyeTap devices and the VideoOrbits Gyroscopic Head Tracker*. In a desire to have the system run faster with an ultimate goal of real-time (30 frames per second), the VideoOrbits implementation was optimized to use modern computer graphics hardware to do computational vision versus traditional image synthesis.

This demonstrates the feasibility of “active” and “shared” mediated reality environments that a wearable computer user can interact with in near real-time. In the future, it would be beneficial

to bring the active and shared components of the mediated reality together so that not only could one interact with one's environment or see another's, the user could also interact with another environment, through seeing that remote environment as if it were their own. Finally, to have all of this running at real-time would allow for the constant use of such systems.

# Bibliography

- [1] Steve Mann. *Intelligent Image Processing*. John Wiley and Sons, November 2 2001. ISBN: 0-471-40637-6.
- [2] Ivan Sutherland. A head mounted three dimensional display. In *Proceedings Fall Joint Computer Conference*, pages 757–764, Washington, DC, 1968. Thompson Books.
- [3] W. Barfield and T. Caudell, editors. *Fundamentals of Wearable Computers and Augmented Reality*, volume 1. Lawrence Erlbaum Associates, Publishers, 2001.
- [4] University of Waterloo. What is systems design engineering? URL: <http://sydewww.uwaterloo.ca/SystemsDepartment/WhatIsSystems/whatissystems.html>.
- [5] Damien Douxchamps and Dan Dennedy. Coriander. URL: <http://sourceforge.net/projects/coriander/>.
- [6] BlueZ. URL: <http://bluez.sourceforge.net/>.
- [7] Felix Tang. Howto build your own irda sir dongle for linux using the mcp2120. URL: [http://www.eyetap.org/~tangf/irda\\_sir\\_linux.html](http://www.eyetap.org/~tangf/irda_sir_linux.html).
- [8] Microchip. Mcp2120/mcp2150 developer’s kit user’s guide. URL: <http://www.microchip.com/download/lit/pline/analog/interface/infrared/51246a.pdf>.
- [9] Steve Mann. Wearable, tetherless computer–mediated reality: WearCam as a wearable face–recognizer, and other applications for the disabled. TR 361, M.I.T. Media Lab Perceptual Computing Section; Also appears in AAI Fall Symposium on Developing Assistive Technology for People with Disabilities, 9-11 November 1996, MIT; <http://wearcam.org/vmp.htm>, Cambridge, Massachusetts, February 2 1996.

- [10] Bruce I. Blum. *Software Engineering: A Holistic View*. Oxford University Press, New York, 1992.
- [11] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissades. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison Wesley Longman, 1994.
- [12] Mark Weiser. The computer for the 21st century. *Scientific American*, (265):94–104, 1991.
- [13] Roy Want, Andy Hopper, Veronica Falcao, and Jonathat Gibbons. The active badge location system. \* *ACM Transactions on Information Systems*, 10(1):91–102, January 1992.
- [14] Andy Harter and Andy Hopper. A distributed location system for the active office. *IEEE Network*, 8(1), 1994.
- [15] Steve Mann. ‘smart clothing’: Wearable multimedia and ‘personal imaging’ to restore the balance between people and their intelligent environments. pages 163–174, Boston, MA, Nov. 18-22 1996. Proceedings, ACM Multimedia 96; <http://wearcam.org/acm-mm96.htm>.
- [16] Steve Mann and James Fung. Videorbits on eye tap devices for deliberately diminished reality or altering the visual perception of rigid planar patches of a real world scene. In *Proceedings of International Symposium on Mixed Reality (ISMR2001)*, pages 48–55, March 14-15 2001.
- [17] James Fung, Felix Tang, and Steve Mann. Creating mediated reality with hardware accelerated image registration. Submitted to IEEE International Symposium of Wearable Computing 2002.
- [18] B. Horn and B. Schunk. Determining Optical Flow. *Artificial Intelligence*, 17:185–203, 1981.
- [19] S. Mann and R. W. Picard. Video orbits of the projective group; a simple approach to featureless estimation of parameters. TR 338, Massachusetts Institute of Technology, Cambridge, Massachusetts, See <http://hi.eecg.toronto.edu/tip.html> 1995. Also appears in *IEEE Trans. Image Proc.*, Sept 1997, Vol. 6 No. 9, p. 1281–1295.
- [20] Michael Artin. *Algebra*. Prentice Hall, 1995.

- [21] R. Y. Tsai and T. S. Huang. Estimating Three-Dimensional Motion Parameters of a Rigid Planar Patch I. *IEEE Trans. Acoust., Speech, and Sig. Proc.*, ASSP(29):1147–1152, December 1981.
- [22] Felix Tang, Chris Aimone, James Fung, Andrej Marjan, and Steve Mann. Seeing eye to eye: a shared mediated reality using eyetap devices and the videorbits gyroscopic head tracker. Submitted to IEEE and ACM International Symposium on Mixed and Augmented Reality 2002.
- [23] Jannick P. Rolland, Larry D. Davis, and Yohan Baillot. *A Survey of Tracking Technology for Virtual Environments*, chapter 3, pages 67–112. Volume 1 of Barfield and Caudell [3], 2001.
- [24] W. J. Maclean, A. D. Jepson, and R. C. Frecker. Recovery of egomotion and segmentation of independent object motion using the em algorithm. *Proceedings of the British Machine Vision Conference*, September 13–16 1994.
- [25] Richard L. Holloway. *Registration Error Analysis for Augmented Reality Systems*, chapter 6, pages 183–217. Volume 1 of Barfield and Caudell [3], 2001.
- [26] Chris Aimone, Andrej Marjan, and Steve Mann. Eyetap video-based projective motion estimation assisted by gyroscopic tracking. Submitted to IEEE International Symposium on Wearable Computing, 2002.
- [27] James Davis. Mosaics of scenes with moving objects. In *Proceedings of CVPR*, Santa Barbara, CA, 1998.
- [28] James Fung. *VideoOrbits of an Algebraic Projective Geometry and Comparametric Equations, with Applications in Computer Mediated Reality*. January 2001.
- [29] Steve Mann, James Fung, and Eric Moncrieff. Eyetap technology for wireless electronic news gathering. *Mobile Computing and Communications Review*, 3(4):19–26, 1999.