Architectures for Untethered Augmented Reality Using Wearable Computers

Panagiotis D. Ritsos

A thesis submitted for the degree of

Doctor of Philosophy

at the

Department of Electronic Systems Engineering

University of Essex

March 2006

Abstract

One of the most interesting fields of computer technology is that of Virtual Reality. People are fascinated by being immersed in three-dimensional, computer-synthesised virtual worlds. There are many example applications such as interactive visualisation and representation for entertainment and education, modelling of construction, manufacturing and maintenance processes, architecture, medicine, annotation and simulations for training. One step further is the notion of augmented reality (AR) where, unlike virtual reality where the user sees only virtual worlds, he or she can see both the real and the virtual at the same time.

One of the potential applications of augmented reality is the 3D reconstruction of archæological sites *in situ*, where the user can be immersed while maintaining a composite view of the real and the virtual surroundings. By using an untethered, mobile, body-worn computer with a see-through head-mounted display and equipped with a location and orientation sensors the user can roam in the composite world as if the scene was entirely real.

The research effort described here concerns the construction of such an AR application, centred around the Roman remains in the Gosbecks Archæological Park on the outskirts of Colchester. Two generations of wearable computers have been implemented. The first, similar to earlier prototypes, provided a test-bed for initial, in-the-field tests, in order to prove the concept and gain practical experience. As anticipated, this prototype provided inadequate performance; however the lessons learned influenced the design of a second, more mature platform. The second wearable, designed and built on the experience gained, is a novel prototype with improved processing power and ergonomics, low power consumption and low cost. The prototypes use GPS, the Global Positioning System, for measuring location and a magnetic sensor integrated into the head-mounted display for determining the wearer's direction of gaze.

A novel wearable AR framework was developed to work with both wearables but primarily to exploit the hardware rendering capabilities of the second prototype. The application software was written in C using the OpenGL graphics library for 3D rendering. The framework encompasses optimisation techniques such as view frustum culling and levels of detail in order to improve rendering speed.

The second wearable computer and the framework were used for fairly extensive field testing, in order to determine the accuracy and stability of the position and orientation sensing mechanisms. In addition, the system was assessed in-the-field by users by means of a novel, questionnaire-based user assessment. The assessment investigated the usability of the second wearable running the wearable AR framework, exploring its ergonomics, visual output quality, positional accuracy and stability as well as the sense of presence and overall impression.

The ultimate aim of this research was to ascertain whether wearable AR can be delivered in a form that can be used by the general public for *in situ* guides for archæological sites at a reasonable cost. The findings show that functionality is achievable, though the cost is higher, and the location accuracy lower, than desired.

Acknowledgements

First and foremost I would like to dedicate this thesis to my family, Dennis, Isabella and Eleanna for their support, guidance and encouragement all these years. Many thanks go to my aunt Isabella and my cousin Natalie for their invaluable help and support throughout my stay in the United Kingdom and those lovely London moments they have given me!

Many thanks go to my supervisor, Dr. Adrian F. Clark, for his help, guidance and support through the eight years I've spent in Essex, for giving me the chance to work on interesting projects that introduced me to the world of wearable computing, for offering me a chance to study for my Ph.D., for guiding me through difficult decisions and for being flexible with my very 'relaxed' scheduling habits.

Many thanks also go to Professor Arnold Wilkins and Jo Angouri for their suggestions and help with the design of the questionnaire.

Many thanks to Peter Noakes for his guidance during the early and late periods of my undergraduate study.

Thanks to all my fellow researchers and members of the VASE laboratory. To Neill Newman for his assistance on my first steps in the world of wearable computing, to Mike Lincoln for all his help with all things Linux, to Eddie Moxey for his help on all things digital in the early days of my research and to David Johnston for his ideas and suggestions on graphics and GPS issues, and his witty criticism of my Hellenic-British mix of humour sense.

Many thanks to Alex Isaakidis for his efforts to continue the wearable computing research of our laboratory and Michael Papadopoulos for his friendship and humorous interventions. Also, many thanks to Christine Clark, for those early temple models and for helping me out with my thesis.

A very big thank you goes to all my very close friends, Anna Liva, Nikolaos Rogopoulos, Athanasios Dimitriou, Alexia Giakoubini, Gregory Gotsis, Antonis Gougoulis, Tasos Sarakinos, Maria Boukala, Sotiris Karagiannis, Georgia Ktistaki and Elena Anagnostopoulou for their support, friendship and encouragement through these years, for being there when it mattered most, for helping me relax during my vacation periods and for keeping me company through the net while I wrote my thesis.

Thanks to fellow doctorate students Gregory Dimitriades, Ioannis Tsalamanis, Dimitris Klonidis and Kostas Charamis for their friendship during the year of preparation of this thesis.

Thanks to online communities of MAZA and the PhD United Squad of Knowledge for the relaxing time we had together during two difficult years for me and for those moments of victory!

Last but not least, many thanks to all those people I forgot to mention and have been there for me in various times, all my friends from the University, past colleagues and department administrators and staff.

Contents

List of Figures

List of Tables

Abbreviations and Glossary

3D	Three Dimensional in XYZ
3DOF	Three Degrees of Freedom
802.11	IEEE Standard for wireless LAN technology
802.3	IEEE Standard for Ethernet
AGP	Accelerated Graphics Port
AMLCD	Active-Matrix Liquid Crystal Displays
ANSI	American National Standards Institute
API	Application Program Interface
ATX	The layout and shape of modern motherboards
Bluetooth	Short-range radio communication protocol — IEEE 802.15
bps	bits per second
CF	Compact Flash
CMOS	Complementary Metal Oxide Semiconductor
COTS	Components/Commercial Off-The-Shelf
CPU	Central Processing Unit
CRT	Cathode Ray Tube
DAC	Digital-to-Analog Converter
DDR-SDRAM	Double Data Rate-Synchronous Dynamic Random Access Memory

DGPS	Differential GPS
Doric	Doric shape
DRI	Direct Rendering Infrastructure
DVI	Digital Visual Interface
EBX	Embedded Board eXpandable
ECEF	Earth Centred, Earth Fixed coordinate system
EMI	Electromagnetic Interference
Ethernet	Local-area network (LAN) architecture
FORTRAN	FOR-mula TRAN-slator
FOV	Field Of View
FPS	Frame Per Second
GPRS	General Packet Radio Service
GPS	Global Positioning System
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HMD	Head Mounted Display
HUD	Heads Up Display
I/O	Input/Output
I2C	Inter-IC — bus connecting integrated circuits (ICs)
IC	Integrated Circuit
ISA	Industry Standard Architecture
LAN	Local Area Network
LCD	Liquid Crystals Display

LSI	Large Scale Integration
Microdrive	IBM’s storage product
MP3	MPEG-2 Layer 3
NiMH	Nickel-Metal Hydride
NTSC	National Television System Committee
OLED	Organic Light Emitting Diode
PAL	Phase Alternating Line
PC	Personal Computer
PC/104	IEEE P996 ISA bus specification
PC/AT	Personal Computer/Advanced Technology
PCI	Peripheral Component Interconnect
PCMCIA	Personal Computer Memory Card International Association
PIM	Personal Information Management
RAM	Random Access Memory
RFI	Radio Frequency Interference
RFID	Radio Frequency IDentification
ROM	Read-Only Memory
RS-170	B/W video format used US
RS232	Recommended Standard-232C
RTCM	Radio Technical Commission for Maritime Services
S-Video	Super-Video
SBC	Single Board Computer
SDRAM	Synchronous Dynamic Random Access Memory

SMS	Short Message System
UML	Unified Modelling Language
USB	Universal Serial Bus
VE	Virtual Environments
VGA	Video Graphics Array
VRML	Virtual Reality Modelling Language
WGS-84	World Geodetic System 1984
WiFi	Wireless Fidelity
XML	Extensible Markup Language

Chapter 1

Introduction

People are fascinated by archæology and by the investigation of older civilisations and their cultures. Of great interest is the study of the architectural sites such civilisations created, either in the form of buildings as simple residences or as buildings built for special functions such as temples and markets.

Visitors of such sites want to learn how their ancestors looked and, how they lived their lives, but the proportion of archæological sites where there are substantial remains above ground is small. Many archæological sites have been completely destroyed while others have nothing but scattered remains. These remains need to be preserved and protected because of their great value and irreplaceable nature. Indeed, the curators of archæological sites are faced by a dilemma: they wish to attract interested visitors but need to do so without disturbing any remaining archæology. This usually rules out *in situ* physical reconstructions, though there are exceptions, such as the Parthenon Reconstruction Project[1].

A popular way of helping people visualise the appearance of archæological sites is through paintings and physical scale models. In recent years, these have increasingly been replaced by three-dimensional computer graphic “walk-throughs.” Ultimately however, both of these are unsatisfactory because it is difficult for a visitor to relate the appearance and physical size of the model (physical or virtual) to the archæological remains. A better solution is for the visitor to be able to visualise the appearance of the model as they walk around the site. This is conventionally achieved through labelled perspective views on plaques. However, by exploiting the simultaneous reduction in size and increase in performance of computer technology, one can imagine a system that uses *Virtual Reality* (VR) technology to visualise the ancient buildings as the visitor

[1] `http://www.culture.gr/`

explores the site. This visualisation of computer synthesised information superimposed on real surroundings is known as *Augmented Reality* (AR).

The scenario described in this thesis involves the use of a *wearable computer* with *location and orientation sensing mechanisms* that projects through a *head-mounted display* (HMD) the view of an achæological site *in situ*, as though it was still intact in its original state. As the user roams around the archæological site, the location mechanism detects his or her position and the orientation sensor the direction of view. This information is used to render a 3D model of the building and project it through the HMD, immersing the user in the 3D reconstruction. In order to achieve a successful illusion that the temple is intact on the site, the real surroundings must be visible, implying the use of a see-through HMD. The user needs to move in the virtual buildings as he or she walks around the site in real time. Also, he or she should be able to turn to any part of the building and be able to see it. The 3D model needs to be a complete, detailed and realistic reconstruction of the building and the user needs to be able to explore any part of the model while walking around the area where it was originally located.

The following sections describe the motivation for this research effort (Section 1.1) and provide an overview of augmented reality (Section 1.2) and wearable computers (Section 1.3), briefly discussing the challenges encountered in these two fields. Section 1.4 describes the combination of these two fields as a basis for designing, building and testing a framework for untethered mobile augmented reality. Section 1.5 presents the research questions this thesis deals with and Section 1.6 the author's contribution.

1.1 Motivation

In the past, researchers have produced proof-of-concept demonstrators for this "wearable AR" (Chapter 3). However, those prototypes are bulky, heavy and expensive. The focus of this research is to determine whether equivalent (or, indeed, better) functionality can be achieved using a lighter, cheaper system designed specifically for the purpose. The wearable computer that forms the basis of this visualisation system has been built using the small form-factor motherboards found in current "life-style" desktop personal computers (PC). The display is a colour HMD (Chapter 2). The technologies used for orientation and location sensing are magnetic tracking and the Global Positioning System (GPS) (Chapter 3) respectively. The models are rendered using graphics technologies similar to those encountered in some modern computer games, in particularly those focusing on 3D environments like first-person shooters[2] (Quake, Unreal etc.) (Chapter 5).

Mobile untethered outdoor augmented reality using wearable computers has received some attention from researchers, mainly in the form of tourist guides such as Smart Sight [253] and GUIDE [51] (see also Chapter 3 for further details). Only a limited number of systems employ 3D reconstructions for outdoors wearable AR, presumably due to its demanding technical requirements. Some systems have been implemented using laptop-based prototypes, e.g. Feiner *et al.* [71], Höllerer *et al.* [100] and Piekarski *et al.* [167]. All these follow a similar paradigm, yet the systems developed are heavy and bulky and are not practical for everyday or commercial use. The author believes that these systems, though successful in proving the various concepts of wearable AR, are difficult to duplicate, control and deploy in outdoor experiments in the field. Their weight and size make them difficult to carry, their complexity makes them difficult to construct and their cost is relatively high. On the other hand, a system that is simple to build, relatively cheap, offers increased upgrade potential and is easy to use in the field would make research on wearable AR much more straightforward and practical.

The research described here is centred around the reconstruction of a Roman temple complex, situated south-west of modern Colchester at the Gosbecks Archæological Park. The choice of site was made for historical purposes, bearing in mind the importance of the Gosbecks site both for Colchester and nationally. The history of the Gosbecks site is briefly described in Appendix A.

Gosbecks has some characteristics that reduce some problems while introducing others. Firstly, it is situated in a reasonably flat area (modulo rabbit holes) at the edge of farmland near the brow of a low hill; the nearest buildings are over 500 m away. Hence, almost an entire hemisphere of sky is visible. There are no trees, pylons, or post-Roman remains on the site to interfere with the 3D reconstructions. There are also no visible foundations that the 3D reconstructions have to abut. On the other hand, much of the site has not yet been excavated, particularly the area surrounding the likely grave of Cunobelin; English Heritage[3] is understandably protective of the undisturbed archæology and does not permit the ground to be disturbed, such as by erecting further signage or radio masts.

[2] First-person shooters are computer or video games where the player's view of the game world simulates that of the character, centred on aiming and shooting with multiple styles of weapons.

[3] `http://www.english-heritage.org.uk/`

1.2 Augmented reality

Virtual reality is a research area that has received great attention since the late 1970s. The term *virtual reality* can be defined as "a computer-generated, interactive, three-dimensional environment in which a person is immersed" [15]. Likewise, Augmented Reality (AR) can be defined as:

> A computer generated, interactive, three-dimensional environment which is superimposed on the real environment in order to enhance it and in which a person is immersed, while being able to see both the real and the synthetic in real time.

AR, also referred to in the literature as *Mixed Reality*, is one of the main areas of virtual reality research. In augmented reality, the virtual environment is superimposed on the real or *vice versa*. In VR there is no presence of the real surroundings, resulting in a totally immersive environment, whereas in AR the real environment must be visible.

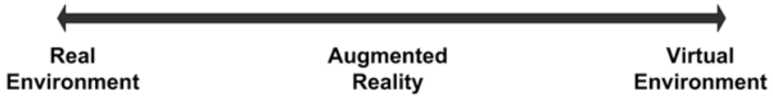


Figure 1.1: Simplified representation of the continuum to virtual environments (adapted from [150])

Milgram *et al.* [150] defined a continuum from real to virtual environments by introducing one more stage, called *Augmented Virtuality* (AV). AR, in contrast to AV, augments information that exists or used to exist, whereas AV augments virtual or fictional information. The technology underlying both AR and AV is similar and, since the research described here deals with the augmentation of a real archæological temple, the two terms are used synonymously as AR. A simplified version of Milgram's continuum shows where AR stands relative to the virtual and real environments (fig. 1.1).

AR suffers from various problems that challenge researchers. Arguably the most important of these is the registration problem. Objects in the virtual and the real worlds must be properly aligned with respect to each other or the feeling of immersion and coexistence (sometimes called the feeling of "presence", i.e. the way a user perceives himself or herself as being present in the computer-generated environment [115, 183, 202]) will be lost. Although registration problems occur in VR, they are harder to detect than in AR. Azuma [18] reports that errors in AR result in visual–visual conflicts because of the use of see-through displays and the simultaneous viewing of real and virtual worlds. In VR systems, the errors result in visual–kinesthetic and visual–proprioceptive conflicts, which are harder to detect. Nonetheless, such conflicts may result in motion sickness (e.g., [159]). In addition, it may be difficult to detect errors in VR systems because the human brain adapts with time to visual information at the expense of other sensory data. This effect, called visual capture [244], increases the tolerance of the human brain to registration errors. On the other hand, in AR the errors are easily noticeable because the virtual and real worlds are visible simultaneously and any discrepancies are always detectable. Generally, the sources of error can be classified in two categories, static and dynamic [18, 19]. Static errors may be introduced from optical distortions in the HMD or other output peripherals, tracking system errors, mechanical misalignments and incorrect viewing parameters, such as the field-of-view or interpupillary distance (see Chapter 2). Dynamic errors occur due to overall system delays.

Atkins [14, 183] defines the overall system lag as:

$$\text{System Lag} = \text{Sensor Delay} + \text{Processing Delay} + \text{Rendering Delay}$$

where the sensor delay is introduced by the tracking subsystem and other sensing of the real environment, processing delay is introduced from the simulation of the virtual environment (object behaviour, scene physics, collision detection etc.) and rendering delay is introduced by the display of the VR scene. Likewise, Azuma [18] defines end-to-end system delay as the time difference between the moment that the tracking system detects position and orientation to the moment when the corresponding changes are displayed. As Holloway [101] pinpoints, overall system lag is the largest source of registration error in modern AR systems.

Bryson [48] describes two types of degradation that occur in VR systems, the update rate and the transport delay. Update rate is the rate at which the visual display is refreshed. Low update rates imply high lag. Transport delay is the time between the user's input and the corresponding representation of that input on the display. Registration errors caused by time delays are apparent only when motion occurs; these registration errors can be quite severe. As stated by Azuma [21, 18], a typical 100 ms end-to-end lag, with a moderate head rotation of 50 degrees per second, results in a dynamic error of 5 degrees. In such a case, in order to keep errors to a fraction of a degree, the end-to-end system delay would have to be no greater than 10 ms; a solution that would be technically difficult. Considerations of this type demonstrate the trade-offs that must be made when designing AR systems.

Lag can affect the user in a number of ways. When it comes to training simulations like flight simulators [78, 146], it may degrade the user's performance, as humans are sensitive to visual discrepancies. Depending on the application, lags between 30–120 ms may have such an effect [92], with a maximum tolerance of up to 200 ms [207]. In some situations, these delays may cause motion sickness, even with small delays in the order of 30 ms [74, 183, 184]. Last but not least, system delays, as well as registration errors, disrupt the aforementioned feeling of presence.

An additional problem in mixing real and virtual world elements is occlusion [82]. In several cases the real and the virtual need to be mixed in scenes where one occludes the other. When virtual information occludes real elements, no problems occur as the display of the virtual information is normally superimposed on the real surroundings. However, in cases where real elements occlude virtual ones, special treatment is needed so that the real appears in front of the virtual. Several researchers have explored different algorithms to overcome occlusion problem in an effort to make their solutions robust, fast and accurate [43, 82, 132, 139].

Ultimately, an augmented reality system is a complex one, incorporating various technologies to achieve its purpose. Computer graphics, advanced optical technology and position/orientation sensors are combined in order to provide a combined view of the real and the virtual environment. Unfortunately, matching the outputs of a computer to the level and realism of the real world is clearly difficult to achieve and, in most cases, impossible with current technology [18]. On the other hand, such great progress has been made in the field of computer graphics and various sensor modalities that demonstrator systems can be implemented.

To re-iterate, an augmented reality system should have the following properties:

- The information must be partially immersive, allowing both the real and the virtual environments to be visible to the user. The higher the realism of virtual information, the better the illusion, the greater the feeling of presence.

- The virtual information must be correctly registered with the real environment in the three spatial dimensions so that no discrepancies appear, as these would be distracting for the user. The virtual information must be in the right place and stable [21]. Angular discrepancies should be a fraction of a degree [17].

- The system must be aware of the user's position and movement within the real environment and update view of the virtual information accordingly.

- The system should respond to the user's movement and change of orientation in real time. End-to-end system delays must be minimal, as they disrupt the feeling of presence. Realistically, and depending on the application, delays can be between 5 ms, virtually undetectable by humans [74], and 200 ms [207].

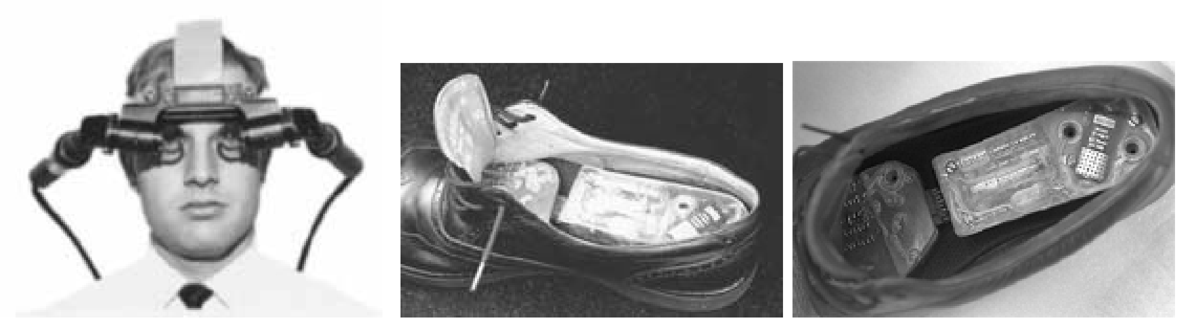
Most current research in AR is indoors, and frequently in specially-instrumented laboratories. Only a limited number of researchers have explored mobile, outdoor, untethered AR systems that use 3D reconstructions, perhaps due to the requirements for such applications in graphics processing power, in addition to the registration problems of location and orientation sensors. Implementing such systems may be the ultimate goal of AR. Allowing users to roam in composite virtual–real worlds freely outdoors can be exploited in fields such as archæology, architecture, computer gaming, military applications and sports training.

1.3 Wearable computing

Wearable computing is a new form of human-computer interaction, based on the use of body-worn computers that are, ideally, constantly powered up and readily accessible by the wearer. Wearable computers differ from personal digital assistants (PDA) and laptop computers in this respect because the latter are not constantly available to the user, requiring initialisation (boot-up) prior to performing any tasks and have a limited operating life between power-ups. Conversely, a wearable computer is, as defined in [142], an untethered computer that is subsumed in the personal space of the user, controlled by the user and has operational and interactional constancy. It has the same features as a desktop computer but in addition it is intertwined with the user, setting it apart from devices such as wristwatches, personal audio players and eyeglasses [142].

Early incarnations of wearable computers were simple portable devices that addressed specific tasks. Although perhaps not considered wearable computers by today's standards, these devices pioneered various aspects of wearable computing. In 1966, Thorp and Shannon revealed their invention of the first wearable computer [226, 225], constructed initially five years earlier, and used to predict roulette wheels. Their device was a cigarette-pack-sized analogue computer with 4 push-buttons. A data collector would use the buttons to indicate the speed of the roulette wheel, and the computer would then send tones via radio to a better's hearing aid. A year earlier, in 1965, Sutherland [218] had created the first computer-graphic-driven head-mounted display[4], using two CRTs mounted beside each ear of a wearer and projecting their image onto the user's eyes with an arrangement of half-silvered mirrors (fig. 1.2(a)). An additional system determined the user orientation of gaze and projected a monoscopic wireframe image of a mid-air floating cube.

Almost ten years later, Farmer and Packard invented a digital wearable computer in a shoe (fig. 1.2(b)) to predict roulette wheels [28], based on a CMOS 6502 microprocessor with 5K RAM and with toe-control and inductive radio communications between a data taker and a better. In 1979, Sony[5] introduced the Walkman, a commercial wearable cassette player which set the scene for today's portable and usually wearable personal media devices such as MP3 players.



(a) Ivan Sutherland's HMD

(b) Farmer's and Packard's shoe-based computer

Figure 1.2: Early incarnations of wearable computers

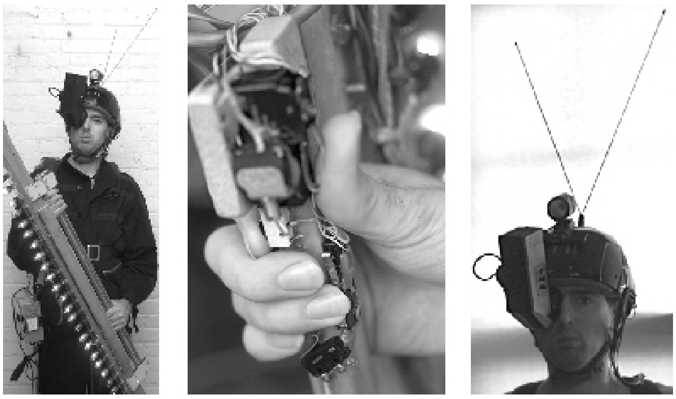


Figure 1.3: Steve Mann's photographer's assistant

[4] Contrary to popular belief, Sutherland did not design the first visually-coupled HMD as Stanton [112] and Heilig [91] patented HMD designs in 1956 and 1960 respectively

[5] `http://www.sony.com`

Nonetheless, wearable computing in the form it has evolved today — miniaturised screen over one or both eyes, body-worn main processing unit and input devices such as push-buttons, chord keyboards and microphones — first appeared in the late 1970s, when Mann built an apparatus to assist him in personal imaging applications [140, 141]. He wired a 6502 computer into a steel-frame backpack to control flash-bulbs, cameras, and other photographic equipment (fig. 1.3). The display was a camera viewfinder CRT attached to a helmet, while input was from seven micro-switches built into the handle of a flash-lamp. The system was powered by lead-acid batteries.

Mann's invention evolved through the years into what he called a 'WearComp' system. In [140] he described it as having three main characteristics:

'Eudemonic' criterion: The apparatus is considered a part of the user and not a separate, added system.

'Existential' criterion: The apparatus is always controllable by the user, either consciously or unconsciously so that it forms an extension to the user's perception, thought and body actions.

Ephemeral criterion: Delays in interaction with the apparatus are imperceptibly small and the apparatus maintains operational and interactional constancy.

Although this definition was presented by Mann for his WearComp system, it was addressed to what he referred to as 'wearable computer' or 'existential computer' [141]. One year later he presented a refined definition [142] of three operational modes for a wearable computer-human interaction paradigm:

Constancy: The computer is always powered up and readily available to interact with the user. The interface between computer and human is based on continuous information flow.

Augmentation: Wearable computing, contrary to traditional computing, is based on the notion that computing itself is not the primary task, as the user is assumed to be doing something else concurrently with the computing. Therefore, a wearable computer is used to augment the user's intellect or senses.

Mediation: The wearable computer, in contrast to laptops and PDAs, can encapsulate us, for example in articles of clothing in direct contact with the body. It is therefore capable of measuring various physiological or body-related quantities such as perspiration, walking/running accelerations, location etc.

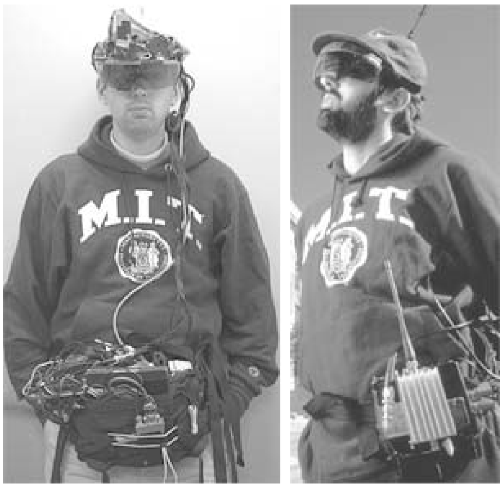


Figure 1.4: The 90s versions of Mann’s wearable computer

Wearable computing has received greater attention since the mid-1990s with various different research prototypes (fig. 1.4) and a number of commercial offerings appearing (Chapter 2). All systems address different flavours of Mann’s problem of ‘personal empowerment’. Although wearable computing uses similar building blocks to traditional computing, they need to be used in a different fashion [212]. The need for observable, yet non-obtrusive, outputs as well as readily-available input interfaces forms one of the major shortcomings in wearable computing, that of the interfaces [157]. Furthermore, operational and interactional constancy are not yet achievable due to the power problems that thwart almost all modern mobile electronic devices that require increased processing power. In addition, due to the special nature of the wearable computers, the design trade-offs have not been explored thoroughly [212, 213].

1.4 Mobile untethered augmented reality and wearable computers

AR in a lab environment can be researched with ordinary computer equipment, e.g. a desktop computer equipped with an HMD and some sensors. Obviously, outdoor mobile AR requires a platform that can be carried with the user.

Wearable computers, described in the previous section, are the ideal platform for this as they are easier and less obtrusive to carry than a laptop, while using a HMD is a normal practice and allows the user to move relatively freely in the virtual environment. The comparison of implementations of AR applications and wearable computer platforms demonstrates the possible convergence of these two fields. Head-mounted displays and hands-free interfaces, real-time processing and location awareness are features met in both fields, indicating that implementing mobile, untethered AR with a wearable computer may ultimately be realised. A definition of a wearable AR computer may be:

> A wearable computer system, equipped with location and orientation sensing mechanisms, that projects through a head-mounted display virtual information overlayed seamlessly on top of the real environment in order to enhance the latter.

The paradigm involves a wearable computer equipped with a location detection mechanism and an orientation tracker to derive the user's position and direction of view. The wearable projects ("augments"), through a transparent head-mounted display, information in the form of graphic elements superimposed on the real surrounding environment (fig. 1.5). The challenges encountered in this scenario involve the aforementioned problems of both AR and wearable computing. The occlusion and registration problems remain the same, although in practice the registration accuracy requirements may not be as stringent as, say, in medical AR [18, 71].

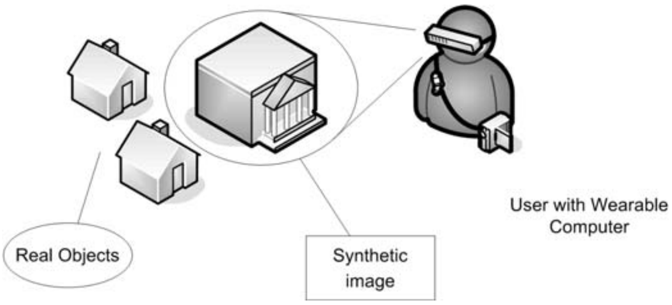


Figure 1.5: Mobile augmented reality

Rendering 3D information on wearable computers is difficult due to the low processing power and lack of hardware support for 3D rendering found in most commercial and research prototypes. Indeed, the only current research prototypes for wearable AR that employ 3D reconstructions are laptop-based systems (described in Chapter 3). Bearing in mind that such wearables involve a compromise between processing speed, power requirements, size and weight, it is quite difficult to produce a system that renders good quality 3D reconstructions and yet remains user friendly, light and active for significant periods of time. Furthermore, these prototypes are typically poor ergonomically, being heavy and cumbersome implementations in excess of 15 kg.

1.5 Research questions

For this thesis the author will attempt to answer the following questions:

Application: To what extend is it possible to produce a complete wearable AR system that allows the general public to experience an *in situ* 3D architectural reconstruction of an archæological site?

Hardware: To what degree is it possible to construct a wearable computer that provides enough rendering performance but is capable of running from batteries for long enough for a person to explore an archæological site in detail? Can the hardware be made small enough and light enough that does not become uncomfortable during this time?

Software: Is it possible to write software for the hardware platform to render a 3D model of the archæological site that is able to operate at high enough frame rate and produce a visually-acceptable appearance and to what extent this can be achieved? Can the orientation and location sensing subsystems offer comparable, if not better, performance than earlier prototypes? To what extent can this software be made independent of the 3D model to be rendered?

Cost: Can the hardware be constructed cheaply enough that a commercial venture could afford to provide (say) several tens of these tour guide systems?

These general capabilities will be quantified in Chapter 4 for the hardware and Chapter 5 for the software.

1.6 Thesis statement and contributions

> *Wearable Augmented Reality prototypes, both in terms of hardware and software do not have to be complex, cumbersome and expensive in order to achieve adequate demonstrable performance, when it comes to real-time in-situ 3D reconstructions.*

Simple, easily manageable and upgradeable, cheap systems can be implemented, while maintaining adequate rendering performance and positional and orientational accuracy. The author believes that, even with a simple software architecture resembling that of modern computer games and employing similar techniques, a simple yet fast, easily manageable and upgradeable software framework for augmenting 3D reconstructions can be devised. To defend the above statement the author makes the following contributions:

- The design and implementation of a novel wearable AR system that allows the *in situ* reconstruction of an archæological site.
- The design and implementation of a novel wearable computer with improved ergonomics, upgrade potential and adequate 3D rendering performance that is to be used for mobile untethered augmented reality demonstrations and experiments *in situ*.
- The implementation of a novel mobile untethered augmented reality software framework which runs on the aforementioned wearable computer platform and augments a 3D architectural model *in situ* with an adequate rendering performance. It uses GPS for location determination and magnetic tracking for orientation sensing. The framework is built on OpenGL.
- The implementation of a 3D model of the main temple of the Gosbecks Archæological Park along with optimisations to increase rendering speed.
- The evaluation of the accuracy of the tracking mechanisms and a questionnaire-based user assessment survey of the overall platform.

1.7 Structure of the rest of the thesis

Chapter 2 provides an overview of the current status of wearable computer technology, with emphasis on the design of research platforms and the associated interfaces. It also provides an overview of some of the main applications of wearable computing.

Chapter 3 focuses on the field of augmented reality, examining some of the research efforts that pinpoint the requirements of AR systems. The second half of the chapter focuses on the research done in mobile AR using wearable computers.

Chapter 4 presents the Romulus wearable prototype that is used as a research platform for wearable AR. The wearable was built from off-the-self components, based on a single board, offering increased graphics rendering capabilities, while remaining simple to build, and easy to upgrade.

Chapter 5 describes the Gosbecks Tour Guide application framework, built on OpenGL and Glut. The application running on the Romulus wearable augments the temple of Gosbecks *in* *situ* and in an untethered mobile fashion. The chapter also describes the optimisation techniques required and the interface mechanisms employed.

Chapter 6 presents the experiments performed concerning the accuracy, the ergonomics and the usability of the system. The evaluation was done in order to assess the usability of the Romulus wearable and the Gosbecks Tour Guide application and their potential for further outdoors AR research.

Finally Chapter 7 presents a summary of this research effort and discusses possible future research.

Chapter 2

The Current Status of Wearable Computing

2.1 Introduction

This chapter aims to describe the current status of the field of wearable computing. A review of the most important research efforts is given, both in terms of hardware and accompanying software. Both systems developed in academic research laboratories and commercial solutions are considered. The purpose of this investigation is to examine the breadth of applications in which wearable computers may be used as well as the practicality of wearable-based services in real-world situations.

Section 2.2 presents the prevailing designs of the wearable computing platforms currently used by researchers with emphasis on the constraints that dictate their designs. Although most of the systems described here are research prototypes, their investigation pinpoints the characteristics of such devices that would eventually render a commercial solution viable and successful. It is the author's opinion that researchers must improve the ergonomics and processing performances of their prototype designs to improve their practicality, as designing a proof-of-principle prototype that merely 'does the job' is only the first step. Systems of higher performance, improved ergonomics and more appealing aesthetics will be paramount to the social acceptance of wearable computing and therefore its commercial development and growth.

Sections 2.3, 2.4 and 2.5 describe respectively the output, input and other peripheral interfaces used in wearable computing prototypes. One of the main problems in wearable computing is the quality and ergonomics of the interfaces [224]. Interfaces for wearable computing is one of the fields that requires extensive further research [157]. Developments in optics, such as advances in organic light emitting diode (OLED) and active-matrix liquid crystal displays (AMLCD), which led to the introduction of higher-resolution HMDs with more compact designs than earlier implementations, increase the quality of the output of the HMDs [209]. In addition, unconventional mechanisms such as voice commands simplify input [157], albeit with additional processing requirements. In the future, personal wireless networks such as Bluetooth should help design elegant systems [211], reducing cable clutter and weight, similar to hands-free mobile phone headsets [41].

Section 2.6 describes commercial wearable systems. The purpose of this review is to demonstrate that most commercial systems are designed to provide low processing power, appropriate for mainstream wearable computing tasks but inappropriate for 3D reconstructions. Usually these systems are based on low-power CPUs of limited multimedia capabilities. Nonetheless, such prototypes are popular in industrial maintenance and the military, demonstrating that wearable computing has matured enough to be used in ‘the real world’ and not merely exist as a research field.

The military was an early adopter of wearable computing. However, as Mann states [142], the evolution of consumer electronics is arguably surpassing the technological sophistication of military electronics. Section 2.7 describes such military applications, giving an overview of the most important wearable computing systems developed for various programmes. Both research prototypes and commercial examples are provided, demonstrating the functionality and potential of wearables in the battlefield.

Section 2.8 attempts a closer investigation of some of the most relevant applications, services and research topics. The aim of this investigation is to present the nature of services that wearable computing provides and the mechanisms deployed. Similar to hardware, the nature and type of services have a direct impact on the ergonomics and efficiency of wearable computers. These services need to be robust and efficient, not merely proving the concept or principle. Nonetheless, both hardware and software efficiency is quite often directly dependant on the available technologies of desktop systems since much of the hardware is common. This investigation aims to show these dependencies.

Since this thesis describes a wearable AR application, mobile augmented reality using wearable computers is investigated in greater detail in Chapter 3, showing that this is one of the most demanding wearable applications. Current research projects and applications are described in detail, both in terms of the hardware platforms used and the software practices followed.

As has already been discussed, augmented reality is an application that can truly exploit the features of a wearable device. Starner *et al.* [221] point out that the closer association of the user with a wearable computer allows the latter to 'see' what the users see. Furthermore, by sensing the users context in terms of location and orientation, appropriate tasks can be performed autonomously [213]. These features can be useful in wearable AR scenarios. Nonetheless, wearable computers are generally performance-limited due to their portable, low-power nature, imposing further challenges [212] beyond those encountered in conventional AR.

2.2 Wearable computing architectures

2.2.1 Form factor, processors and prototype systems

At the beginning of the 1980s, progress in integrated circuit (IC) technology allowed the introduction of single-board computers. Large scale integration (LSI) logic chips for CPUs, memory and displays allowed the shrinkage of these boards, making them popular for systems that required increased processing power. The trend continued, resulting in the so-called embedded Single Board Computer (SBC) market. The fact that the PC architecture was becoming a worldwide *de facto* standard generated the interest to create embedded systems that were compatible with PCs.

Wearable computing is directly related to the SBC market as most prototypes are based on such boards. Their small size, along with PC compatibility, makes them ideal for such systems, allowing developers to use readily-available components — often referred as COTS (commercial or components off-the-shelf) — and software. The most popular form factors for wearable computer prototypes are:

- modular building blocks such as the PC/104 and PC104/Plus boards described further below;
- all-in-one SBCs such as the LittleBoard/EBX form, expandable with plug-in modules.

Tin Lizzy and PC/104

By far the most popular architecture for a wearable computer prototype is the 'Tin Lizzy', first introduced by Starner and Platt at MIT in 1992–93.[1] The computer was constructed from PC/104

[1] `http://www.media.mit.edu/wearables/lizzy/lizzy/`

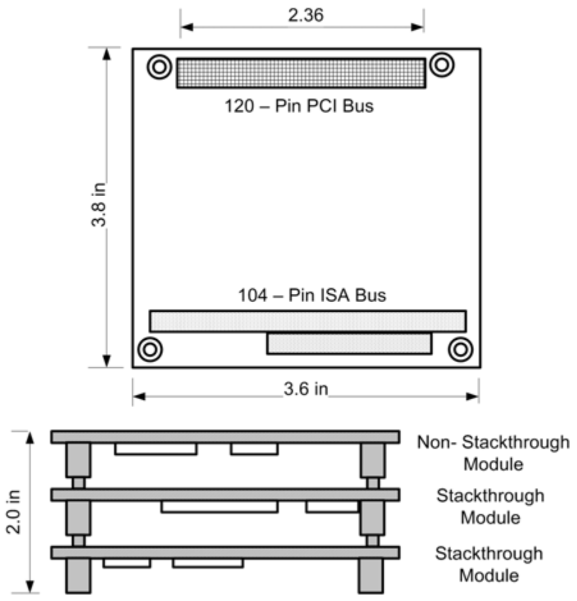


Figure 2.1: The PC/104-plus module and stack configuration

boards designed for industrial control. The size of these boards allows fairly small-sized computers to be constructed. The boards can be connected on top of each other to form any required configuration with relative ease. The only requirement is to connect the on-board interfaces to the appropriate connectors on the enclosure of the wearable.

PC/104 standard boards [161, 162] are similar in functionality to standard PC motherboards and PCI/ISA expansion cards. Their size is 3.6 × 3.8 inches (fig. 2.1). They have a typical consumption of 1–3 W per module and are stackable through special bus connectors. The basic modules used in a wearable are a CPU board, usually with a 486 or Pentium[2] class processor, a VGA card and a hard disk. Many systems include a PC card adapter — usually for Wireless LAN PC cards — and additional I/O cards for interfacing to application-dependent peripherals. Recent CPU boards come equipped with VGA and Ethernet (802.3) adapters, making the size of the wearable even smaller.

The availability of such boards allows relatively powerful systems to be constructed. Many researchers favour this configuration due to their ease of construction and customisation, along with the fact that the system is usually relatively small in volume. Furthermore, PC/104 modules are rugged enough to be used in a mobile fashion. Nonetheless the weight of these prototypes can be quite high, sometimes around 5 kg, especially if more than two modules are used and a metal enclosure is used for carrying and cooling.

[2] http://www.intel.com/

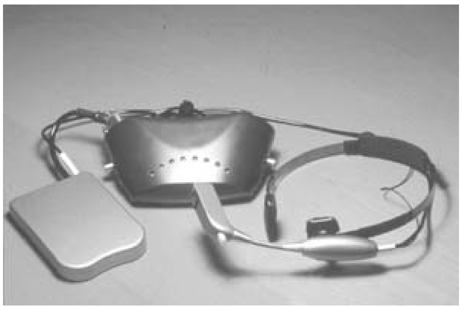


Figure 2.2: The Spot wearable from Carnegie-Mellon University

Other embedded boards

The fact that PC/104 boards are used for industrial control means that some of the requirements of wearable computing are not met by them, especially features such as 3D graphics acceleration. Many systems have been built around other available types of boards such as the aforementioned EBX form factor [23] or the Plug'n'Run board [216]. The benefits from the use of such hardware may be improved multimedia, while in some cases the systems can be smaller and lighter than their PC/104 counterparts. In most cases the power requirements are similar to PC/104 and they provide about the same processing power, though usually with a higher cost.

On the other hand, some embedded boards are based on low-power processors such as the StrongARM or Transmeta's Crusoe.[3] That is the case of the StrongARM-based LART boards[4] for example, which may result in a lighter wearable suitable for personal information management tasks or applications that do not require significant graphic processing.

An example of a wearable based on a low-power SBC is the Spot wearable (fig. 2.2) from Carnegie-Mellon University [68].[5] Spot is based on a custom-made StrongARM SA-1110/1111 SBC with an impressive potential for peripheral connectivity, including USB, RS232, 802.11b and DVI interfaces. With a weight of 270 g without batteries and a power consumption of 470 mA at 12 V, it appears to be an excellent solution for many wearable computing tasks. Indeed, Spot could be easily used in almost all wearable computing applications that do not require accelerated 3D graphics.

Careful design and attention to detail make Spot a promising choice as it combines low-power performance with a large number of interfaces, small size and low weight. Moreover, Spot demonstrates that custom-made solutions can result in successful designs. However the

[3] `http://www.transmeta.com/`

[4] `http://www.lart.tudelft.nl/`

[5] `http://www.wearablegroup.org/hardware/spot/index.html`

time and resources required for the development and implementation of such a custom-made system is available to only a few universities and research centres.

MIThrill and modular architectures

Similar to systems based on low-power, custom-made boards are those that implement a modular, reconfigurable and customisable architecture. These are based around a number of basic boards, usually called *cores* or *nodes*, which are interconnected using serial communication protocols such as USB or I2C.

An example of a wearable computer based on low-power, modular boards is the MIThrill 2000 (fig. 2.3) and its 2003 variant, both developed in the MIT Media Lab [234].[6] The wearable is constructed from a number of boards interfaced together forming a pair of 'networks' called the *Body Network*, which connects the main computing cores and a wireless bridge, and the *Body Bus*, which connects peripherals and sensors. The main components are a Linux-based core based on the ipEngine from Bright Star Engineering,[7] an Intrynsic CERFboard[8] based on a StrongARM 1110, a wireless 802.11b bridge and a custom-made, microcontroller-based data acquisition platform.

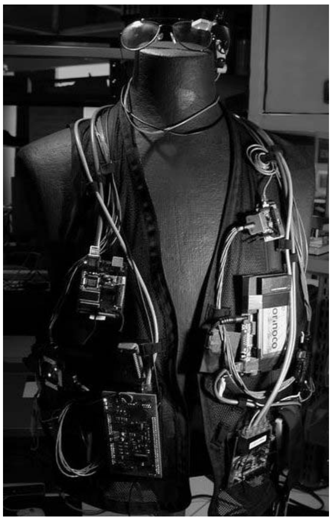


Figure 2.3: The MIThrill wearable from MIT

This system is quite different in its approach from most of the development platforms used up to now and is one of the few efforts to explore different hardware architectures for wearable

[6] http://www.media.mit.edu/wearables/mithril/index.html

[7] http://www.brightstareng.com/

[8] http://www.intrinsyc.com/

computing. The difference from most current designs is that the system is based on an array of light-weight, low-power cores interfaced with an ethernet connection rather than the more common PCI/ISA bus. According to its developers, this system allows the connection of a number of different sensors and input modalities through its *Body Bus*. They argue that this method will allow simpler connection of custom peripherals, allowing further customisation. This approach was followed in order to introduce a unified method of interfacing peripherals. MIThrill 2000 was intended for context-aware applications, so the number of inputs that may be interfaced to it is high. The fact that the system is based on low-power, small-sized boards may resolve the problems with the power-hungry, bulky and heavy systems currently used.

An extension of MIThrill 2000 was the WearARM project [10] from the ETH-Zurich. This was a low-power, high-performance wearable computing platform based on a SA1110/SA1111 chipset, SDRAM, flash memory and VGA DAC. A number of ports, such as PCMCIA and USB, have been implemented. WearARM was eventually to be integrated with MIThrill to provide a complete wearable computer platform, very different from the Tin Lizzy architecture.

The MIThrill 2000 developers argue in favor of some of their choices in implementing the first MIThrill system, yet pinpoint some of its deficiencies as well [234]. The choice of Linux for operating system as well as the use of I2C for connecting various computer cores may have been successful. The author, though, remains sceptical of the overall approach because it results in a complicated system with excessive, possibly redundant, wiring due to the dedicated Ethernet connection for each core. The developers of MIThrill confirm that implementing the system required fair experience with hardware, as well as high cost, a fact that made the system less successful than anticipated. This led to a new version, the MIThrill 2003 [234].

The use of a low-powered CPU core is something that may prove popular with developers. Most of today's Personal Digital Assistants (PDAs) — and many mobile phones — use such processors with success. On the other hand, one must bear in mind that these lightweight, modular systems are mostly used in applications that do not require high processing power. Their limited power consumption allows small, light-weight systems to be built; but they would not prove adequate for demanding applications such as AR using 3D reconstructions.

Personal digital assistants (PDAs) and handheld-based wearables

MIThrill 2003 is based on the same body bus concept as its 2000 counterpart but its main processing core is a Zaurus SL-5500 PDA running Linux, interfaced with the body bus via RS232.

The Compact Flash (CF) slot on the PDA allows the use of various peripherals such as Bluetooth and WiFi (802.11b) cards. Various sensors are interfaced to the body bus, either as stand-alone microcontroller-based devices or through a 'Hoarder' Sensor Hub [234], an extension module that implements combinations of sensors such as digital accelerometers, biometric sensors and infra-red tag readers. MIThrill 2003 combines the approach of a PDA-based primary core with the modular reconfigurable secondary modules used in the original MIThrill.

There are also examples of wearables based solely on PDAs. Such an example is Ackermann's second generation iPAQ-based wearable computer [1]. This prototype is based on a Compaq (now HP) iPAQ PDA interfaced to an HMD and a chord keyboard. The wearable is a dual-boot (PocketPC/Linux) device with Linux on a compact flash card or a microdrive in a PCMCIA jacket, running a full-featured Linux distribution tailored for the iPAQ.[9] PDA-based systems may be easier to work with, require relatively little hardware expertise and are fairly robust. On the other hand, interfacing peripherals such as HMDs and input devices may be difficult and is directly dependant on the PDA's expansion capabilities. Ackermann's implementation uses a MicroOptical HMD connected to the serial port via a serial-to-NTSC/PAL converter and a chord keyboard connected on the same serial port.

PDA-based systems are light, small in volume and easy to carry. Their ever-increasing processing power is adequate for simple tasks. On the other hand, they are usually limited in interfacing options, particularly with output devices such as HMDs.

Laptops and backpacks

Another popular approach is the use of laptops as the main processing unit. Bearing in mind that today's laptops can be small yet relatively powerful, many researchers opted to use them as they usually offer increased stability compared to custom-made configurations, plus the aforementioned benefit of multimedia capabilities. For applications that involve intense graphic processing, the use of laptops with hardware-accelerated 3D rendering is an effective solution.

These systems are mostly of higher processing power than 'Tin Lizzy' systems. The problem though, as in many 'Tin Lizzy' implementations, remains that the systems are bulky and heavy. The author's experience with a PC/104 prototype (Appendix B) has shown that a bulky and heavy system hinders the in-depth evaluation and testing of wearable AR applications. Similarly, laptops used in hand-held fashion are tiring to use and do not allow the user's hands to remain

[9] http://www.iptel-now.de/

free. Backpack systems, such as Tinmith [167] or Columbia University's Touring Machine [71] are usually heavy — well in excess of 15 kg — and bulky. Nevertheless, these systems are used in most wearable augmented reality applications and the most relevant examples are reviewed in detail in Chapter 3.

It is apparent that all the architectures described above have advantages and disadvantages. When high processing power is required, most researchers prefer laptop-based configurations. For AR applications, wearables equipped with powerful graphics cards are being used in backpacks to provide the required rendering performance. On the other hand, implementations such as MIThrill, Spot and PDA-based systems are used for less demanding tasks where ergonomics and low power consumption are more important. In the latter case, the number, quality and nature of the interfaces of a wearable can be important, leading researchers to use dedicated mechanisms for incorporating different input modalities. MIThrill's Hoarder expansion hub and Spot's impressive range of interfaces are such examples.

It is also apparent that the nature of the application governs the configuration of the system. When demanding processing is required, researchers have to compromise the ergonomic features of a wearable — such as weight, volume, size — and power requirements. When input modalities increase they add to the overall weight of a wearable computer and may increase obtrusiveness. Likewise, when there is no need for high processing performance, low-power CPUs and motherboards are being used, thus increasing the operating life between battery recharges and decreasing factors such as weight, volume and size. The use of a small USB hub to accommodate many peripherals is a simple solution that does not affect significantly the size and volume of a system; nonetheless power is affected when the input devices are not self-powered. On the other hand, carrying separate battery packs for peripherals can be a nuisance.

2.2.2 Power requirements

Wearable computing suffers from the same main problems as other mobile technologies. One of these problems, probably the most important from the user viewpoint, is power consumption. The value of a wearable system is directly related to how often a user or a developer needs to recharge its batteries. There are other methods of powering a wearable device such as kinetic energy [129, 49] or solar panels [210] but these are likely to be adjuncts to, not replacements for, battery power for many years.

Hahn *et al.* [144] classify the power requirements of wearable devices in three categories:

- batteries for the main wearable computer unit;
- batteries for the various discrete electronic devices such as sensors or HMDs;
- batteries for various sensors distributed in the user's local environment used in conjunction with sensors connected to the wearable unit.

The first two categories are the most important in wearable computing. In the first category there are batteries that power the wearable computer's main unit, usually similar to those found in modern laptops. These batteries can be either custom-made as in Birmingham's WearCAM [60], or camcorder batteries such as Essex's Rome MK-I [157]. The operating life of a battery is directly dependent on the load introduced [86]. In the case of Rome MK-I, the operating life was less than an hour with a configuration of four PC/104 cards, powered from a pair of Duracell 12V camcorder batteries. The WearCAM wearable with a single PC/104 card powered from a custom-made 12V NiMH battery cell is claimed to have an operating life of eight hours. Although the batteries are probably not similar in terms of capacity, the difference in operating time is indicative that the configuration of the main unit has an impact on the battery life.

Powering smaller devices and peripherals is simpler, e.g. a handheld GPS unit, powered using AA batteries. Powering such devices from the main battery is not necessary and, if avoided, results in less wiring. Furthermore, most of these devices draw currents in the order of a few mA, compared to main units which usually draw 2–3 A.

Small electronic devices can also be powered using kinetic energy. Using piezoelectric elements, Kymissis *et al.* [129] derived power (averaging 0.23 W) from the bending of shoe soles and the impact of the heel while walking. This is claimed to be sufficient to power small RFID tags that transmit the wearer's position every 3–6 steps. Similarly, a rotary generator can be mounted on the heel of a shoe, providing twice as much power compared to the piezoelectric elements. The problem with such an approach is that the generator is difficult to integrate with a shoe due to its size and fragility.

Likewise, solar cells can be used to supply low-powered electronic devices with power. The MIT Locust location system [210] uses 9V solar cells mounted under overhead lights in labs to power small infra-red beacons which serve as location tags for wearable computers. The solar cells charge when the overhead lights are on, resulting in a virtually maintenance-free system, at least in terms of power.

2.2.3 Operating systems

One of the factors for researchers in the area of wearable computing is what operating system they will use on their research prototypes. Microsoft Windows[10] is the option some researchers prefer because of its popularity and the availability of applications. However, Windows is not particularly flexible when it comes to support for special-purpose hardware and requires large amounts of memory usually not found in wearable computers. Furthermore, Windows do not offer the flexibility of other operating systems when it comes to working with alternative graphical user interfaces (GUI); a requirement that some researchers have noted as windows-based interfaces are inappropriate for wearables [157]. Last but not least, development of new applications as well as replicating wearables based on Windows can be arguably fairly expensive compared to some alternatives.

The main alternative is the Linux operating system, currently quite popular among wearable computing researchers. Linux is a free Unix-type operating system, developed under the GNU General Public License[11]. The source code for Linux and a large number of applications is freely available to everyone. Various distributions are available from different suppliers, with differences in the installation procedure, applications, upgrade and update system, and support.

Linux is much more lightweight than Windows, significantly more customisable, supports most development languages and is generally easier to manipulate. Of great importance among researchers is the community support on technical issues which results in easier and faster development of the drivers and software usually required for specialised hardware.

Another alternative that may be used in the future but has received little attention to date from researchers are operating systems used in mobile ’phones and PDAs such as SymbianOS[12] and Windows variants such as WinCE.[13] These operating systems could be easily adapted for use with wearable computers designed for simple personal information management (PIM) tasks. Commercial products such as Xybernaut’s POMA[14] wearable PC make use of WinCE. Some of Nokia’s mobile phones[15] are based on SymbianOS, offering simple tasks such as PIM applications, gaming and Internet access along with standard mobile telephony applications. In the case of WinCE, application development is constrained to the use of Microsoft Embedded Visual

[10] http://www.microsoft.com/

[11] http://www.gnu.org/

[12] http://www.symbian.com/

[13] http://www.microsoft.com/

[14] http://www.xybernaut.com/

[15] http://www.nokia.com/

Studio suite whereas SymbianOS supports Java, C++ and Visual Basic. Nonetheless, support for development of applications for such systems is limited, much poorer than for Linux.

In the author's opinion, Linux is probably the best choice of operating system for a wearable due to its flexibility and support for development through the open source community. Linux can be used for 'heavy' Augmented Reality applications as well as other, more 'lightweight' tasks with equal success.

2.3 Output peripherals

One of the main problems in the area of wearable computing is the use and type of the interfaces employed. Bearing in mind the aspiration for non-obtrusiveness and hands-free operation, input and output mechanisms different from the standard monitor, keyboard and mouse need to be devised [157]. The most popular solutions are outlined below.

2.3.1 Head-mounted displays

As far as output is concerned, the use of a head-mounted display seems to be the most popular solution. HMDs are probably the trademark of virtual reality applications due to the wide number of uses they find in this field. Usually they are *fully-immersive*, and thus relatively obtrusive for use with a wearable. Monocular HMDs such as Micro-Optical's viewers[16] or Tekgear's M1[17] are considered better solutions for most wearable computing applications because they allow the user to be more conscious of his or her real surroundings and are more socially acceptable than bulkier, fully-immersive models. Both occupy a relatively small proportion of the user's view while maintaining at the same time a screen of adequate resolution. Binocular and biocular see-through models are preferred in AR applications. In binocular models each eye sees a different image, whereas in biocular models both eyes see the same image. Monitor-based configurations are also encountered in AR [18], using a similar approach to the Head-Up Displays (HUDs) encountered in military aircraft [88, 89, 112]. These are not reviewed here because they cannot be used in wearable fashion.

The optical characteristics of HMDs include the field of view (FOV), image quality, luminance, focus, exit pupil and eye relief. Image quality, often called resolution, is measured in dots, pixels or the angular subtense of a single pixel [148] and dictates the fidelity of the image.

[16] `http://www.microopticalcorp.com/`

[17] `http://www.tekgear.com/`

The field of view is the angle subtended from the HMD as viewed from the user [148]. It is measured in degrees and usually is defined in both horizontal and vertical (and sometimes in diagonal) angles.

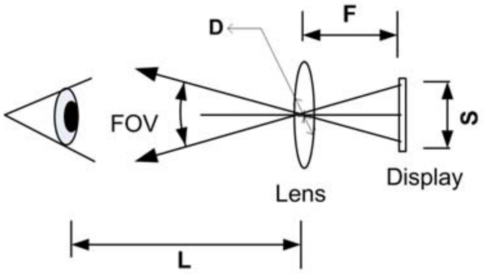


Figure 2.4: Field of view geometry (adapted from [148])

In monocular HMDs the optical field of view (fig. 2.4) can be calculated from:

$$FOV = 2\arctan\frac{S}{2F} \tag{2.1}$$

where:

FOV = field of view

S = size of display (in or mm)

F = focal length of lens (in or mm)

In an extreme case, if the eye is too far back from the lens the optical field of view may not be viewed completely due to the small diameter of the lens. In this case the field of view (also referred to as the *instantaneous field of view* for aircraft HUD [148]) is calculated from Equation 2.2. For a well-designed HMD the instantaneous and optical FOV should be the same [148].

$$FOV = 2\arctan\frac{D}{2L_e}, \quad D < L_e\left(\frac{S}{f}\right) \tag{2.2}$$

where:

FOV = field of view

D = diameter of lens (in or mm)

L_e = eye distance from lens (in or mm)

The calculation of the field of view becomes more complicated in biocular and binocular HMDs. Figure 2.5 shows the different FOVs that occur in a binocular display. It is claimed by Melzer and Moffit [148] that it is preferable to rotate the optical axis of each display inwards so that, with the addition of properly-aligned lenses, the left eye sees the right part of the scene and the right eye the left part. The authors term this configuration a ‘convergent’ binocular display. They also claim that a similar approach should be taken for single lens biocular displays.

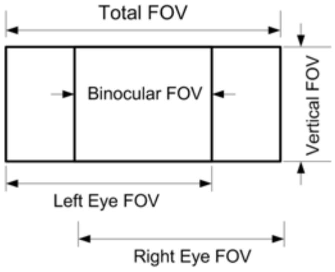


Figure 2.5: Different FOV that occur in biocular and binocular displays (adapted from [148])

In most cases the field of view and the image resolution are inversely proportional to each other, apart from the more expensive HMDs which combine both high resolution and wide field of view [19, 148]. In the case of AR, an HMD needs to be transparent so that the real and the virtual are both visible. In such cases luminance, the photometric quantity that approximates the visual sensation of brightness, also plays important role [148]. Most see-through HMDs are not designed to be used in bright daylight; their brightness is adjusted for room lighting (see Chapter 3). More expensive models have adjustable brightness, allowing customisation according to the operating conditions. Luminance is dependant on the source luminance (CRT or LCD display) and the efficiency of the optical system used to project the image to the user (optical combiners etc.) [148].

Focus is also a factor that affects the quality of an HMD. As shown in figure 2.4, the lens of the display system is fixed to a specific focal distance from the user’s eye. However, focus can be adjustable in an HMD, using a mechanism that changes the distance of the lens relative to the user’s eye. This allows near-sighted and far-sighted users to use the HMD [148]. The relationship between the image and the lens position is:

$$D_i = \frac{dF}{F - d} + l_e \tag{2.3}$$

where:

D_i = distance from eye to virtual image

d = distance from lens to display

F = focal length of lens

l_e = eye distance from lens

When the virtual image is at the focal point of the lens, $F = d$ and $D_i = \infty$. When d is larger than F the distance is negative and the image lies between the lens and the eye and is real. The above value for the distance between the eye and the virtual image can be translated to optical power [148]:

$$OP = -\frac{1}{D_i} \tag{2.4}$$

where:

OP = optical power (measured in dioptres, m^{-1})

D_i = distance from eye to virtual image

Another parameter which affects HMD performance is eye-relief, the distance between the eye and the nearest HMD component, as well as the exit pupil, the position where the eye needs to be in order for the HMD display to be fully visible [148]. If the eye is outside of this area, the user will not be able to see the full display. The relationship between eye relief, exit pupil, lens focal length and diameter and display size is:

$$E = D - \frac{L_e S}{F} \tag{2.5}$$

where:

E = exit pupil size

D = lens diameter

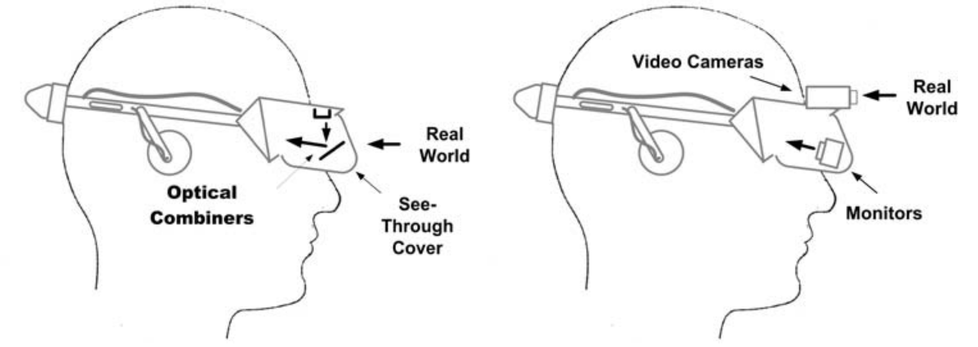
S = display size

F = focal length of lens

L_e = optical eye relief distance

Equally important for an HMD is the unit's weight and the ability to accommodate different head sizes. In general, the lighter the HMD, the easier it is to be worn for longer periods [123]. Adjustment harnesses allow for comfort and different head size and shape. Weight distribution also affects comfort [123]. If the weight is concentrated on one side, the HMD may be unpleasant to wear for long. Last but not least, the cost of an HMD can be high. Expensive HMDs can provide higher resolution, more colour depth and a wider field of view, which together can be very effective for AR and VR. The price of high-end products can run well beyond $\pounds 10,000$.

There are two major categories of see-through HMDs, optical and video [17]. Optical models rely on the use of partially-reflective and transmissive optical combiners. These allow the real world to be visible, while at the same time reflect an image of the computer-generated images (fig. 2.6(a)). The amount of light that passes through these combiners affects the view of the real world and it is a matter of careful HMD design. In expensive models this can be adjustable. The result is a composite view of real and synthesised worlds. Video see-through models on the other hand use cameras to record real world imagery, which is then projected along with the computer-generated scene through miniaturised monitors or LCDs in front of the user's eyes (fig. 2.6(b)).



(a) Optical see-through HMD

(b) Video see-through HMD

Figure 2.6: Head mounted display configurations

Optical see-through models offer simplicity of implementation and therefore reduced cost, a non-degraded view, in terms of resolution, of the real world and no eye-offset. Eye offset effects occur when the cameras of the video see-through HMD are not in the same position as the eyes, or the distance between them is not the same as the interpupillary distance [17]. On the other hand, video see-through models offer more options in the composition of virtual and real worlds and registration methods. The latter can be achieved by applying image processing techniques to 'align' the virtual to the real in order to produce a composite image. Delays between the real and the virtual worlds can be partially synchronised by delaying the video stream of the real world and matching it to the virtual [19]. Obviously, for that to work efficiently the delays in generating the virtual image have to be small, otherwise the overall delay of the composite image may be unacceptable.

An interesting type of optical HMDs are Virtual Retinal Displays (VRDs) [125, 227][18]. The VRD creates images by projecting modulated, low-power laser light directly onto the retina

[18] `http://www.hitl.washington.edu/research/vrd/`

of the eye. The viewer has the illusion of seeing a 14-inch monitor at two feet distance. The claimed advantages of VRDs are a large FOV ($> 120°$), resolution approaching that of the human eye, full-colour quality images, adequate brightness for outdoor use, true stereo with depth modulation and low power consumption. VRDs can be both see-through and non-see-through. Microvision Inc.[19] has the exclusive license for commercial virtual retinal displays and produces the Nomad HMD, described in the next section.

2.3.2 Commercial head mounted displays

The descriptions that follow review HMDs that can be used for wearable computing, that is see-through designs; non-see-through HMDs are obviously not useful for wearable computing. Both HMDs that have been used in wearable computing systems and recent commercial solutions, not necessarily encountered in wearable computers, are reviewed.

Tin Lizzy's Private Eye

One of the first wearable computing output devices used was the Private Eye, a head-up, 720×280 monochrome display which produced an 80×25 character screen. The Tin Lizzy design initially used the Private Eye for output of text-based information. Text-based applications, such as accessing email with Pine, command shell, or the Emacs-based Remembrance Agent [187] reviewed in Section 2.8, run adequately through the Private Eye.

Tekgear M1

In recent years, one of the more popular output devices for wearables has been the M1 monocular HMD from Tekgear (fig. 2.7). The M1 provides a medium-resolution (320×240) grayscale display and consists of an HMD and belt-pack drive electronics assembly. It can be connected to a standard VGA or NTSC video stream via a standard 15-pin VGA connector.

The M1 is a popular display with researchers because it is easily customisable and has a low cost ($\approx \pounds 300$) compared to alternatives. It has been used in various research prototypes (e.g., [157]). Tekgear provides enough information about the driver unit for it to be modified to operate at 5 V instead of 12 V and the drive electronics can easily be integrated into enclosures and backpacks. Many researchers have customised the M1 into various configurations, the most no-

[19] `http://www.microvision.com`

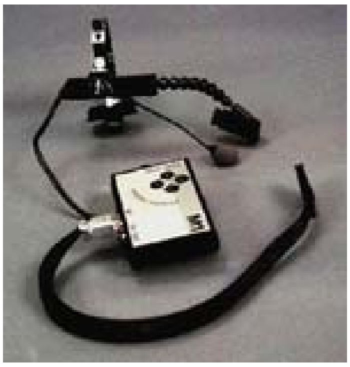


Figure 2.7: TekGear's M1 personal display

table being Steve Mann's Wear7 sunglasses.[20] Most of these custom systems involve integrating the M1 with fashionable spectacles.

Virtual i/O 'I-glasses!'

Of similar popularity to the M1 is the Virtual i/O I-glasses, one of the first fully-immersive, binocular HMDs intended for consumers (fig. 2.8). The I-glasses provides two colour LCDs, and features a three degrees-of-freedom orientation tracker. It is capable of displaying either the same image in both displays or different images in front of each eye, for stereo output. The HMD is driven from a 640×480 VGA input at 60 Hz. Its effective resolution is not certain though, because of the limited literature and the fact that different values are quoted on the manufacturer's website and in the manual. It is certainly sub-VGA, yet of satisfactory resolution for prototype work. The HMD uses a separate driver unit of 15×2×10 cm size and is powered by a 9 volt supply. The I-glasses have appeared in many prototypes, most notably the Touring Machine [71], despite having been discontinued.

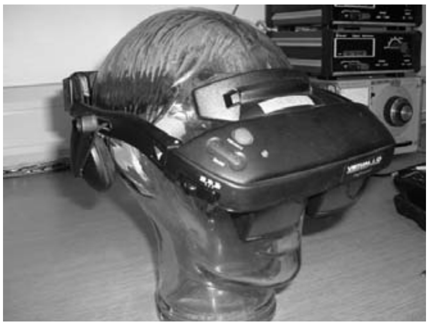


Figure 2.8: The Virtual i/O "I-glasses!"

[20] http://about.eyetap.org/

Sony Glasstron and clones

Many developers now use the Sony[21] Glasstron series of biocular HMDs (fig. 2.9), as in the case of the Tinmith project [167]. Although not equipped with a head orientation tracker like the I-glasses, they provide superior (yet monoscopic) image quality at 800×600 pixels (SVGA). They have a $28°$ horizontal field of view and provide stereo audio output via headphones and can work with an SVGA or an NTSC/PAL source. Sony has discontinued the Glasstron series but a few models can still be found on the market.

Figure 2.9: Sony Glastron HMD

Several manufacturers, such as Canon and Olympus, have produced similar designs to the Glasstron, mainly directed towards gaming and entertainment users. These HMDs, due to their high cost and limited consumer demand, have been discontinued. Some models, such as Canon's GT270[22] or Daeyang's i-Visor models,[23] can only be found in the Asian market in scarce supplies.

Micro-optical

Probably the least obtrusive and most elegant HMD currently available is Micro-optical's monocular, eyeglass-mounted Eyeglass Display System [209], making it a popular solution with researchers and developers. The highest specification model in the range offers a 640×480, 24-bit colour display format, accepts standard VGA input and is particularly light-weight (40 g approximately). The system is easily mounted on almost any adult eyeglass frame (fig. 2.10), with the option of left or right eye configuration. It has a diagonal field of view of $20°$. They have been used in various research efforts, including MIThrill [234] and Ralf Ackermann's iPAQ wearable computer [1]. The highest model in the range, the SV-6, is priced at $\approx$ £1200. Micro-optical also offers various binocular models through OEM programmes but these are not see-through.

[21] http://www.sony.com/

[22] http://www.canon.jp/

[23] http://www.personaldisplay.com/english/default.html

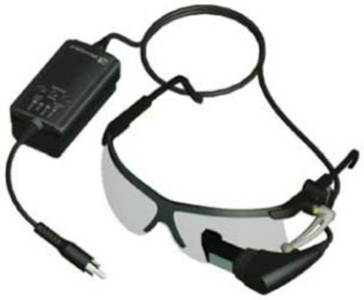


Figure 2.10: Micro-optical's Eyeglass display system

Saab AddVisor

The AddVisor 150[24] is a transparent high-resolution, binocular HMD (fig. 2.11). It uses two independent full colour 1280×1024 pixels (SXGA) Liquid Crystal on Silicon microdisplays. It can be used for mono and stereo viewing. Images can be superimposed on the environment with a 35% or a 100% visibility. It has an adjustable field of view of $46°$ up to $60°$. It uses a 24 bit input and has optional audio output. Although high in specification, its cost is high ($\approx \pounds 54,000$).

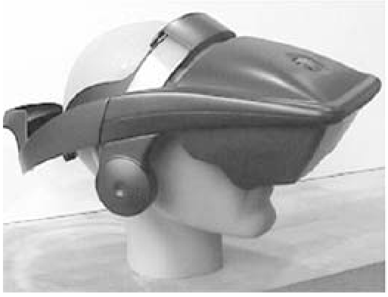


Figure 2.11: The Saab Addvisor 150 HMD

Kaiser Proview

Kaiser's Proview XL40/50 is a high-resolution (1024×768) monochrome-green, see-through HMD, with integrated head-tracker and audio headset (fig. 2.12). It has a $45°$ diagonal field of view (100% overlap) adjustable to $55°$ and adjustable luminance. It uses a pair of reflective combiners with independent optical channels and has a weight of 800 g. As in the case of the Saab AddVisor, the impressive specification has also a high price ($\approx \pounds 31,500$).

[24] http://products.saab.se

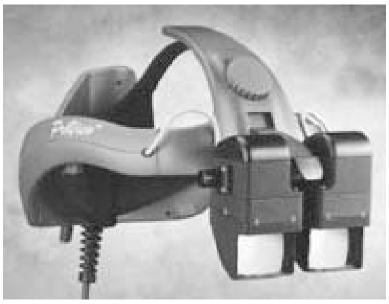


Figure 2.12: The Kaiser Proview XL40/50 HMD

Microvision Nomad

Nomad is a see-through, high-resolution, monochrome-red, monocular HMD, also encountered in the Nomad wearable computer (fig. 2.13). It has a 800×600 SVGA resolution, user adjustable luminance and focus depth, and a field of view of $28°$. It weighs 500 g and accepts analogue VGA input, using the aforementioned VRD technique for projecting images onto the user's retina. The outside world is visible while operational with a 45% visibility. It has no audio output and it is of moderate cost ($\approx$ £$4,450$).

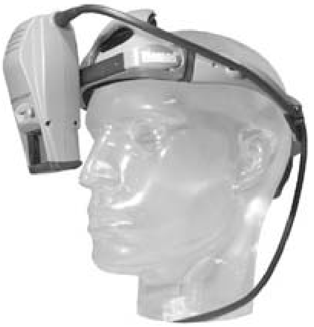


Figure 2.13: The Nomad HMD

Shimadzu DataGlass2/A

The Shimadzu[25] DataGlass2/A is a see-through, monocular, colour HMD, with an option for left or right eye configuration (fig. 2.14). It has an 800×600 maximum resolution and a diagonal field of view of $30°$. The DataGlass2/A supports 24-bit VGA input and does not require an external power source as it is powered from the host computer's USB port. No audio input or output is included. The brightness is software adjustable. It has been used with the Poma

[25] http://www.spi-inc.com/dataglass/

wearable from Xybernaut and in IBM prototypes. It is one of the lightest HMDs available with a weight of 80 g. It is currently available only in USA and Canada ($\approx \$2,400$).

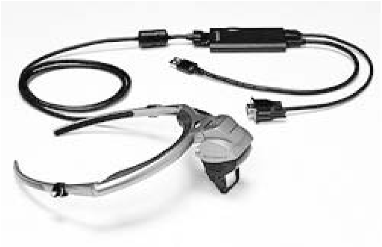


Figure 2.14: The Shimadzu HMD

Datavisor SeeThrough

Nvision's [26] series of HMDs include a see-through version of their flagship stereo, colour HMD, the Datavisor Hires. This model, named Datavisor SeeThrough, has a 1280×1024 maximum resolution with a 78° horizontal field of view and adjustable image plane focus (fig. 2.15). It accepts VGA input of 24-bit colour depth and it is available with an option for motion tracking. No audio is included and the cost is fairly high ($\approx £14,000$).

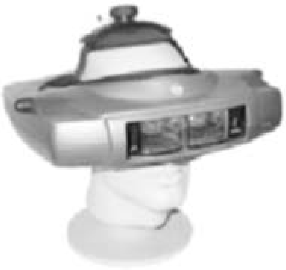


Figure 2.15: The Datavisor See-through HMD

EyeTop

Ingineo [27] offers a range of HMDs addressed to the personal and private video entertainment industry. From their complete range, the Eyetop Classic is most appropriate for wearable computing applications (fig. 2.16). It is a monocular HMD with a video resolution of 320×240, a 16° field of view, a weight of 60 g and accepts 16-bit colour depth input. Both right and left

[26] http://www.nvisionindustries.com/

[27] http://www.ingineo.net/

eye versions are available. Although its design is more compact than most of the aforementioned displays its low resolution is somewhat restrictive. It offers audio output by means of headphones and it is of relatively low cost ($\approx \pounds 300$).

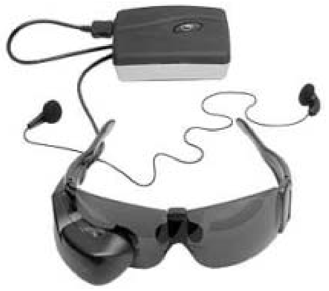


Figure 2.16: The EyeTop HMD

Trivisio ARvision

Trivisio's ARvision 3D HMD is a full stereoscopic video see-through HMD, using two colour cameras with NTSC/PAL and optional VGA output signals (fig. 2.17). Focus, interpupillary distance and convergence are adjustable. The field of view is approximately 40° diagonally. The HMD uses a pair of 800×600 SVGA displays for output and its weight is 230 g. However, a separate control unit of 480 g is required. The control unit offers adjustable brightness and contrast, an electronic zoom and alternative inputs (NTSC/PAL/S-video/VGA/composite video). The cost, bearing in mind the specification, is moderate ($\approx \pounds 5,000$).

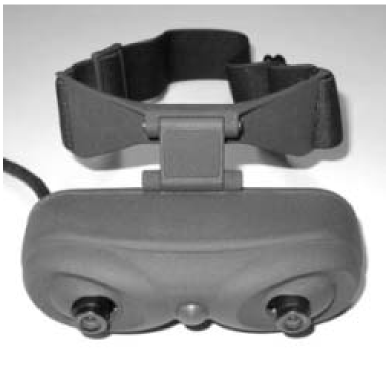


Figure 2.17: The Trivisio ARvision HMD

2.3.3 Non-HMD outputs

Although HMDs are the most popular solution for wearable computers, researchers have also used simple LCD screens such as those found in PDAs to provide output in a more conventional way. Commercial wearables such as Xybernaut's models come with such displays. Of course, handheld LCD screens are not useful in AR applications, unless they are being used as secondary displays as in the case of the Fujitsu Stylistic 2003 used in [100].

Audio output can be as important as visual. Audio, usually played through small earphones, is easily implemented in hardware and the only requirement is the existence of a sound card. Substituting visual output with audio is useful in cases where the user needs to be focused on his or her surroundings, such as while jogging or driving. The author believes that augmentation of the real environment can be aided from the inclusion of virtual world sounds that increase realism and therefore the feeling of presence.

2.4 Input peripherals

Input peripherals impose more problems than output ones in the context of wearable computers. The reason is that replacements for the keyboard and the mouse prove to be relatively difficult to use for most people. Most unconventional input peripherals are based on the use of one-handed chord keyboards, where a sequence or combination of keys produces a character. Chord keyboards often include mouse functionality as well, either in the form of a trackball or a tilt mechanism. They are used by stenographers, with much higher data entry rates achieved than with QWERTY keyboards, yet they are fairly difficult for most users to master [157]. The use of macros may improve input speed and ease of use.

The application considered in this research does not require explicit user input from devices such as a keyboard or a mouse. The system monitors only the user's position and orientation and accordingly updates the virtual world. No text input or pointer manipulation is required as the application is initiated immediately after booting and the user does not manipulate objects of the virtual world. In spite of this, the brief review that follows focuses on input peripherals that have been used with wearable computers in the past; it is not a complete assessment of the possible solutions that could be used. The purpose is to present the most popular devices used by researchers today in order to complete our consideration of research prototypes.

2.4.1 Twiddler, Bat and chord keyboards

Probably the most popular chord keyboard is the Twiddler from Handykey.[28] The Twiddler generates characters by pressing combinations of keys, has four control buttons and a tilt mech-

[28] http://www.handykey.com/

anism to simulate mouse functionality and is connected to a serial port. Its second version, Twiddler2, can be connected to a PS2 or USB port (the latter via an adapter) and uses an IBM Trackpoint instead of a tilt mechanism for mouse functionality (fig. 2.18).

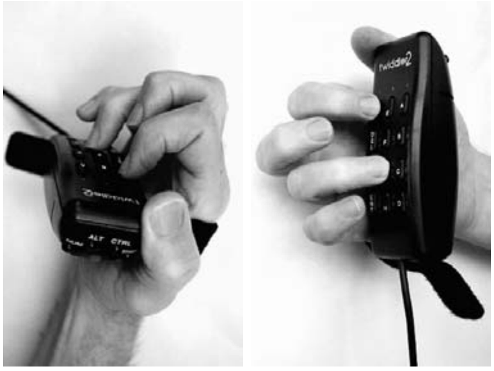
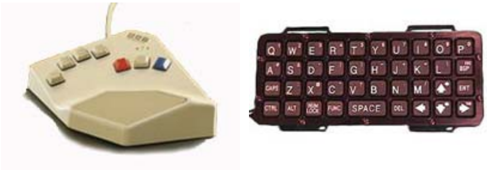


Figure 2.18: Handykey's Twiddler2 chord keyboard

Another popular chord keyboard is Infogroup's BAT chord keyboard.[29] It uses seven keys to provide the same functionality as a standard keyboard (fig. 2.19(a)). Although it has been quite popular with researchers, the author believes that its lack of mouse functionality may be a problem. Many users are accustomed to use a pointer to select menu entries — although the nature of menus arguably should be different from those currently encountered on desktop machines [157, 189]. Although a pointer can also be controlled with keystrokes it is more intuitive for most to use a mouse. Using keystrokes to control the pointer, or even select menu entries directly, may require some training and a familiarisation period that many may not be willing to spend.



(a) (b)

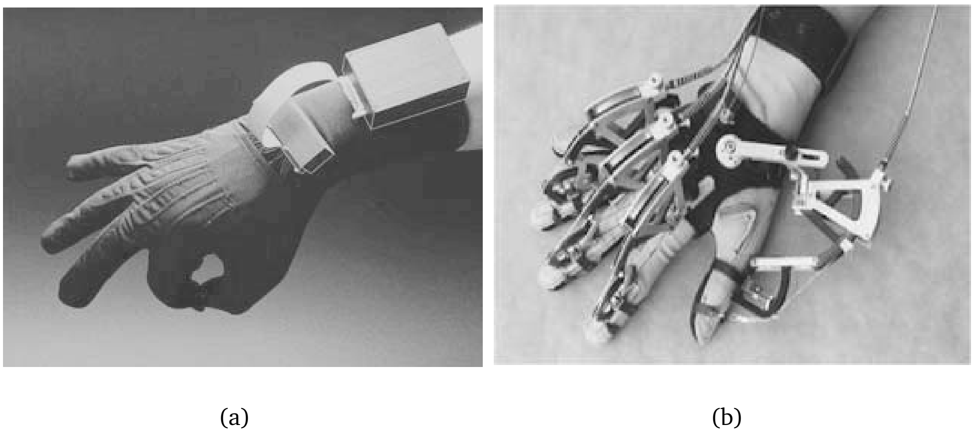
Figure 2.19: Infogrip's BAT chord keyboard and a wrist-mounted keyboard

A somewhat simpler approach to a conventional keyboard is the use of a small-sized, wrist-strapped QWERTY keyboard (fig. 2.19(b)). These are considerably easier to use but occupy both hands. Furthermore, there might be a need for a pointer as in the case of the BAT.

[29] `http://www.infogroup.com/`

2.4.2 Haptic and gesture Input

Apart from chord keyboards, which emulate desktop keyboard functionality, alternative methods have been used as inputs in wearable computing. One of these methods is the use of data gloves and pinch gloves (fig. 2.20). Pinch gloves measure hand movement and finger contact, simulating keystrokes, whereas data gloves also detect finger flexion. Data and pinch gloves can be used in VR/AR environments to manipulate virtual objects or to replace mouse functionality in AR environments, as in [167, 169].



(a) (b)

Figure 2.20: Examples of data and pinch gloves

Data gloves, like the DataGlove 16 from 5Dt[30] (fig. 2.20(a)), are interfaced to a serial or a USB port of a computer and come with specialised driver software. Of course, if the user needs to perform tasks in his or her environment using his or her hands, data gloves may be restrictive.

2.4.3 Audio input

Ultimately, audio input may be more important to a wearable user than a keyboard. As with audio output, dictating commands to a wearable "personalises" the computer and is much closer to human nature, desirable for a machine that constantly interacts with its user and remains attached to him or her through the day. On the other hand, it may be socially unacceptable in certain cases to use audio: an example could be a cinema, where a user of a wearable might like to input comments about a movie to the wearable for his or her personal record. Dictating the comments is not an option in such a case.

[30] http://www.5dt.com/

Speech recognition is feasible these days. IBM's ViaVoice[31], which runs on both Windows and Linux platforms, is popular among researchers. The Linux version has been discontinued but various developers have used it, for example [158]. By issuing simple commands as requests, tasks such as note-taking or checking the time can be performed. Nonetheless, speech recognition requires an effort from the user to gain familiarity and is affected significantly by ambient noise.

2.4.4 Pen/touch screens

Apart from chord and small keyboards, researchers have used pens and touch screens as means of input. An example is the Mitsubishi Amity used in [71] and the Fujitsu Stylistic 2003 (fig. 2.21) [100]. In those cases, the hand-held computer is connected to the main processing unit, a backpack with a laptop, and serves as a menu selector using an integrated stylus, in a manner similar to a PDA.

Although pen-based input devices are far easier to master than chord keyboards, they occupy both hands of the user. Hence they are only practical in applications where the user is not required to manipulate any objects in his or her real environment, such as doors, handles etc.

Figure 2.21: The Fujitsu Stylistic 2300

2.5 Other peripherals and capabilities

Networking support is essential for any wearable to prove useful in the modern computational environment. This can be provided most easily by the use of a wireless LAN. IEEE 802.11 is the prevailing protocol and can achieve high data rates (≈ 10 Mbps using TCP data rates). 802.11 can easily be used in wearable computing by means of a commercially-available PC card or USB adaptor. These allow the wearable to access the Internet and run standard network programs

[31] http://www-306.ibm.com/software/voice/viavoice/

such as FTP and ssh. The main practical limitation of such adaptors is their effective range. Some products may have a reduced effective range (10–20 m) inside buildings due to reflection and attenuation by the building infrastructure. Ideally, a wearable with a wireless LAN card or USB adapter could switch base stations as the user roams within the building in a fashion similar to the GSM mobile telephony cell protocol.

Another solution for networking access could be the General Packet Radio Service (GPRS) which allows information to be sent and received across a mobile telephone network [116]. GPRS has a theoretical maximum speed of approximately 170 kbit/s, adequate for simple communications tasks, however download and upload speeds of 57.6 kbit/s and 14.4 kbit/s respectively are more realistic (for GPRS classes 8 and 10). This can be implemented using commercial PCMCIA adaptors with built-in GPRS transceivers. A subscription with a mobile telephony provider is also required.

For short range wireless communications, Bluetooth[32] may also be used. Bluetooth is an industrial specification for wireless personal area networks (PANs). It allows the interconnection of devices like PDAs, mobile phones, laptops and PCs via short-range radio frequency links. Versions 1.1 and 1.2 can reach speeds of 723.1 kbit/s whereas Version 2.0 can reach 2.1 Mbit/s. Various commercial USB adaptors are available and could be used to equip a wearable with Bluetooth connectivity.

Furthermore, a wearable computer's peripherals may include location and orientation sensors. Location sensors can be classified in two categories, indoor and outdoor. As far as outdoors is concerned, by far most popular and simple to use is the Global Positioning System (GPS). The fact that GPS does not work properly indoors (see Chapter 5) requires the introduction of alternatives. Two that have gained popularity are the Active Badge [238], developed by Olivetti Research Laboratory, and the Locust wearable positioning system [210]. Both are based on small microcontrollers and infra-red or ultrasound transceivers. The Locust system is based around a general-purpose board called iRX.[33] It is connected to a wearable through a serial port and can inform the user of his or her location within a building, to assist in location-based tasks. Each room can be equipped with a Locust which constantly transmits an ID. A wearable equipped with a similar Locust receives the room ID and replies with its own ID. This way, both the user and the room are able identify each other.

[32] http://www.bluetooth.com/

[33] http://www.media.mit.edu/~r/projects/picsem/

2.6 Commercial wearable computer systems

This section describes commercial wearable systems systems, comparing them where appropriate to the research prototypes described above. Generally, commercial wearable computers are low-processing power solutions, directed mainly to industrial uses but not able to cope with 3D reconstructions, as stated by Zhong *et al.* [256]. In some cases commercial systems are also used in military applications, like Microvision's Nomad.

2.6.1 Xybernaut

Xybernaut[34] is one of the leaders in commercial wearable computer systems and holds a number of patents relating to wearable computers. Their range of wearables uses Microsoft operating systems and is capable of accommodating various peripherals through protocols such as USB, IEEE 1394 (Firewire) and RS-232. The MA V model is based on a 500 MHz Mobile Celeron processor with 8 MB video memory and options for external hard disk and compact flash cards. It uses the XyberView HMD, which comes in both monocular and binocular varieties. The Atigo M and Atigo T models are touch-screen based computers, based on Intel's XScale processor and Transmeta's Crusoe running Windows CE. They can be used in conjunction with the MA V wearable.

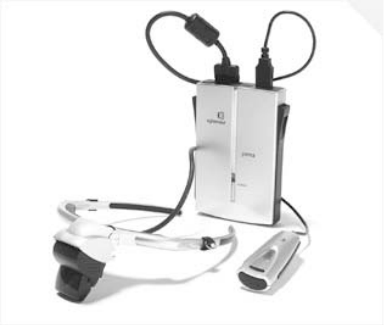


Figure 2.22: The Poma wearable from Xybernaut

One of the latest systems they have developed, in association with Hitachi, is the POMA (fig. 2.22). It has a 128 MHz RISC processor, 32 MB of RAM, a CF slot and a USB port. Poma supports up to 1 GB microdrives, wireless modem cards and uses Microsoft's WinCE operating system. It is equipped with a 640×480 monocular HMD and weighs 310 grams. It is aimed at users who require PDA-like functionality in the form of a wearable computer.

[34] http://www.xybernaut.com/

2.6.2 Microvision

Microvision, Inc.[35] offers a wearable system called NOMAD, based on a Windows CE embedded computer and an augmented reality vision system (fig. 2.23). The wearable is equipped with a 802.11b wireless link, has integrated touch-pad and buttons and supports Flash ROM memory. The output system is the aforementioned Nomad VRD. Nomad has an operational life of 8 hours and it is intended for industrial and military applications. Microvision markets the system as a technician's aid for one-screen display of automotive and industrial blueprints and as a military system offering situation-awareness, tactics display and medical records (for field-medics).

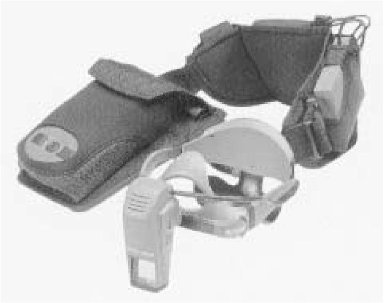


Figure 2.23: The Microvision Nomad wearable

2.6.3 ViA

ViA's wearable computer,[36] the ViA II PC, is based on either a 166 MHz Cyrix processor or a 600 MHz Crusoe. It is belt-mounted and has serial, USB and PC Card interfaces (fig. 2.24). A choice of Microsoft operating systems is available. It is quite light compared to research prototypes, weighing 625 g.

Figure 2.24: The ViA II wearable

[35] http://www.microvision.com

[36] http://www.via-pc.com/

2.6.4 Charmed

Charmed[37] offers a wearable computer called CharmIT in two options, a 266 MHz Pentium or an 800 MHz Crusoe-based system. The first version uses a PC/104 CPU board whereas the Crusoe version uses a Mini Embedded SBC board. Expansion is provided through the PC/104 and MiniPC buses respectively. The Crusoe version features audio, infra-red ports and 256MB of RAM; the Pentium version has a lower specification. Both systems support USB, a number of serial ports and Ethernet. It is noteworthy that the system is based entirely on off-the-shelf components and it is similar in construction to many research prototypes, contrary to other commercial systems.

2.6.5 IBM

IBM Corporation used 'teaser' television adverts to present its wearable prototype to the public, although it did not provide any further information. Information appeared on various IT websites, presenting a compact and powerful solution.[38] The IBM prototype (fig. 2.25) is based on a Pentium 233 MMX processor, has 64 MB RAM and a 680 MB hard drive. Its size is 26×80×120 mm and it weighs 400 g. Input is by a two-button mouse replacement connected to the serial port.

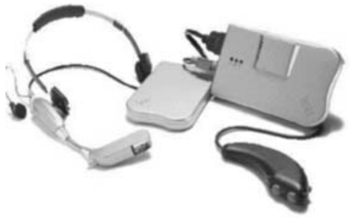


Figure 2.25: The IBM wearable Prototype

2.7 Wearable computers in the military

2.7.1 Soldier's Computer

One of the potential fields of application of wearable computing is the military as the notion of personal empowerment [142] should be very useful in the battlefield. Indeed, in 1989, James

[37] http://www.charmed.com/

[38] http://www.spectrum.ieee.org/publicfeature/oct00/wear.html

Schoening, a research analyst for the US army, envisioned the use of a small, wearable computer equipped with an HMD that would assist soldiers in the battlefield [257]. In 1990 Schoening and Zieniewicz presented the Soldier's Computer, a 386-based computer with integrated radio, weighting 10 pounds (fig. 2.26). The main unit was based on a single-board computer with expansion cards and 16 MB of RAM. A serial connection to a GPS receiver provided location information which was displayed to a helmet-mounted Private Eye display (see Section 2.3). The success of the system with military officials led to the commencement of the Soldier Integrated Protective Ensemble (SIPE), shifting the wearable design to an open-system bus architecture [257].

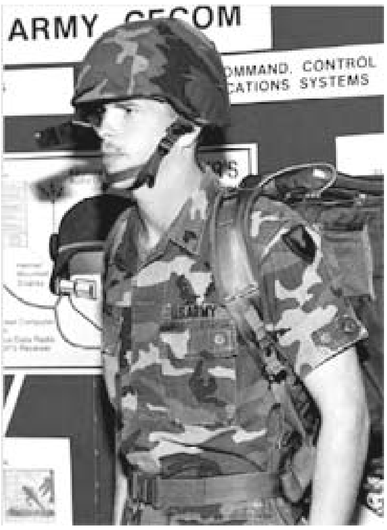
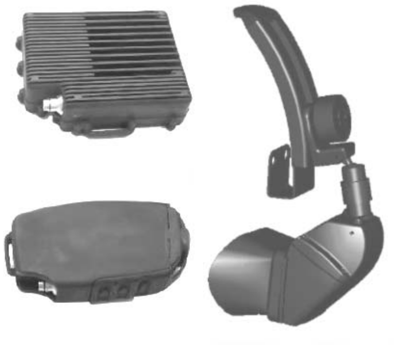


Figure 2.26: The Soldiers Computer wearable (from [257])

The Soldier's Computer included a GPS receiver, a data radio (FM), a video capture system based on a Sony XC-999 camera, a digital compass, a miniature color camera, a video controller subsystem, an HMD, a power supply subsystem, wiring harnesses, and packaging. Using an application developed in C, the system would allow the display of maps, maintain situational awareness, prepare written field reports, capture reconnaissance photographs and provide access to military reference material. One of the features that made the system useful was the link between the Thermal Weapon Sight (TWS) on the weapons and the helmet display. It allowed soldiers to use their weapons while hidden behind corners or trenches, exposing only their hands. However, the system was heavy and the speed of capturing and transmitting photos over the 9,600 bps link was very slow, requiring over a minute sometimes per picture.

2.7.2 Land Warrior

Building on the experiences gained from the development of Soldiers Computer, the US Army initiated a new project, Land Warrior, in 1993 [154, 257]. Land warrior aimed to offer situational awareness, communication capabilities, various sensory modalities and reconnaissance systems while remaining lighter, smaller and lower-powered than its predecessor. In 1999 the development of the first version, Land Warrior Version 0.6, began, based on commercial off-the-shelf components. It was subsequently tested with the 82nd US Airborne Division in a set of training scenarios involving location determination after parachute landing, transmission of the soldier's location in urban attack scenarios and a night mission [257]. The tests showed a series of shortcomings, yet the system proved popular with the soldiers.



(a) (b)

Figure 2.27: The Land Warrior wearable components

These tests assisted in the development of the first field version (Version 1.0), called the Land Warrior Initial Capability. The Pentium-based computer subsystem ran Windows and weighted 820 g, consisting of a main computer unit based on a PC/104 board, flash memory, video board, all packaged in the computer subsystem box (fig. 2.27(a)). The box had connectors for power, USB, and IEEE 1394 connections. The Land Warrior application software was stored on flash, along with maps, field manuals, and system information. A secondary unit, the communications and navigation subsystem, was based on a Intel StrongArm processor running Windows CE (fig. 2.27(a)). The helmet subsystem consisted of an HMD, headphones and microphone. The HMD was an 800×600 pixel, full-colour OLED display viewed through a high-efficiency plastic prism encased in a protective housing (fig. 2.27(b)).

During tactical movement and contact, the soldier primarily used the system to view his or her location, other friendly unit locations, and had his or her heading superimposed on the area map. Version 1.0 also incorporated power management to improve system life. The weapons subsystem (WSS) had a firearm-mounted Daylight Video Sight (DVS) and Thermal Weapons Sight (TWS) for sighting. A grip on the weapon's stock had buttons that let soldiers make calls, switch between sighting systems, capture images, and locate targets. The navigation system used a GPS receiver with an antenna mounted on the left shoulder and a magnetic compass heading sensor.

2.7.3 Battlefield Augmented Reality System (BARS)

BARS is a research project of the Advanced Information Technology branch of the US Naval Research Laboratory with Columbia University [134]. The system consists of a wearable computer, a wireless network system and a see-through Head Mounted Display (HMD) with orientation tracking. The user's view of the environment is enhanced by superimposed graphics onto his or her field of view. The system, following the paradigm of laptop-based wearable AR, is based on a Dell Inspiron 7000 Pentium-II 366MHz laptop. An Ashtech GG24-Surveyor GPS receiver and an InterSense IS300Pro inertial tracker provide location and orientation information, while a FreeWave Radio Modem, capable of transmitting 115 kbits/s over long ranges, provides communication capabilities. A Sony Glasstron Head-Mounted Display (HMD) is used for output (fig. 2.28). The system essentially is a refinement of the Touring Machine [71]. The purpose of the system is to provide information to a soldier in urban environments, such as wireframes of buildings with marked positions of enemy and friendly forces, or other annotations such as tactical information. The system uses an information database of elements, such as buildings, military units etc. A filtering system is used to control the information that is displayed, depending on the user's location.

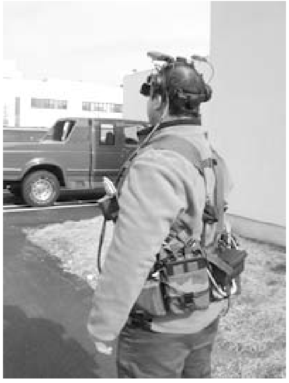


Figure 2.28: The BARS wearable

2.7.4 Commercial wearables in the military

Apart from the aforementioned programmes, commercial wearables are also employed in field testing in the military. The Nomad system is used by the Stryker Brigade Combat Team (3rd Brigade of the 2nd Infantry Division),[39] to provide armoured vehicle commanders with location awareness information through maps displayed on the system's HMD, as well as enabling them to see simultaneously the horizon and superimposed information from their vehicle tracking system [84]. Xybernaut's Atigo T/HB is also used in the U.S. Army Apache attack helicopter maintenance at Fort Campbell[40] [147]. The system is tailored to support maintenance personnel, uses a powerful satellite communications unit and enables global access to maintenance resources and live video feeds.

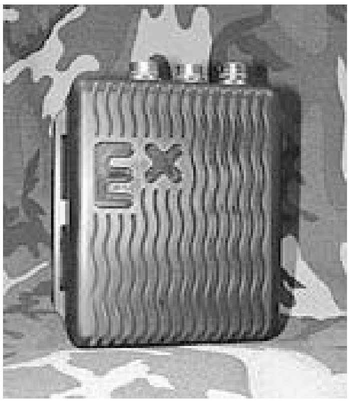


Figure 2.29: The Test Bed wearable computer

Exponent[41] is a company working with the U.S Army's Natick Soldier Center[42] in the development of a wearable, to be used as a test prototype in various electronics projects (fig. 2.29) as part of the Objective Force Warrior (OFW) Science and Technology (S&T) program. The Test-Bed Wearable Computer (TBWC) uses a Transmeta Crusoe 800MHz, has 256 RAM of memory, a standard IDE interface and it runs Windows and Linux. It has inputs comparable to modern laptops, with 2 USB ports, a 802.11b wireless LAN interface, VGA/DVI output, audio output and four serial ports.

Last but not least, Thermite from Quantum3D[43] [52, 122] is a powerful wearable system fea-

[39] http://www.strykernews.com/

[40] http://www.campbell.army.mil/

[41] http://www.exponent.com/practices/techdev/TBWC.html

[42] http://www.natick.army.mil/

[43] http://www.quantum3d.com/products/Thermite/thermite.html

turing a 1 GHz Embedded CPU with 512 MB of DDR system memory and an Nvidia[44] GeForceFX Go 5200 Mobile Graphics Processing Unit (GPU) with 64 MB DDR Memory (fig. 2.30). It supports dual analogue and digital outputs with resolutions up to 2048×1536 in 32-bit colour. It has extensive I/O capabilities including Ethernet, USB 2.0, wireless Ethernet (IEEE 802.11), Bluetooth and a PCMCIA expansion slot. It also provides video-in, video-out functionality with support for NTSC, PAL, S-Video and RS-170A formats. Thermite was developed for the U.S. Army Research Development and Engineering Command (RDECOM)[45] for the RDECOM Distributed Advanced Graphics Generator and Embedded Rehearsal System (DAGGERS) Program.

Figure 2.30: The Quantum3D Thermite wearable computer

2.8 Wearable computing applications

2.8.1 Pervasive/ubiquitous computing

Mark Weiser first introduced the notion of *Ubiquitous Computing* in 1988 [188]. In ubiquitous computing (the more modern term is *Pervasive Computing*), a person interacts with computers embedded in his or her environment. These computers can include location-sensing mechanisms and environment control systems as well as biometric systems or emotion recognition (see also Section 2.8.2).

“Smart rooms” are one application of ubiquitous computing, where intelligence is embedded in an area. By entering a room, the user could command the system to adjust the lighting and heating. Such a system has been implemented by Mann, where sensors were sown into clothing and, by monitoring body perspiration, could determine if the user found the ambient temperature high or low and thus change it through a radio-controlled heater [141]. Interaction with cameras can, in principle, provide further facilities such as face recognition for security.

[44] `http://www.nvidia.com/`

[45] `http://www.rdecom.army.mil/`

Infra-red sensors can determine a user's location and pass electronic messages such as "entrance is forbidden in this room" or "The lab will be closed today for demonstrations".

Wearable computers can be a powerful tool in ubiquitous computing applications. Rhodes, Minar and Weaver [188] argue that there are several problems with privacy and maintenance of personalised information in a ubiquitous computing system. Privacy problems may occur if sensitive information about a user's actions and preferences leaks out to other users. Maintaining information for each user's preferences on environmental sensors, such as settings for heating controllers, would require storage space usually not found in such devices. Wearable computers provide an alternative, by keeping all the important information concerning the user to the user, or rather to the wearable. This way, personal profiles can be passed to each environment by the wearable at the user's will.

Nonetheless, the existence of two or more users in a location may introduce problems in managing the available resources if the personal profiles of the users clash. An example in relation to the aforementioned temperature control system could occur if two users with different ambient temperature preferences entered the same room. Last but not least, some of the resources used may require more processing power or storage capacity than a wearable usually is able to provide (e.g. information databases).

The use of a wearable computer may also solve problems of ubiquitous computing. Hive [188] is an effort to introduce an appropriate framework for such a system. The proposed platform sets out to demonstrate it through the use of distributed agents linked to environmental embedded computers in a peer-to-peer architecture. The demonstration system is a wearable equipped with Locusts [210] which provides location information using infra-red beacons. Hive's agents are autonomous, requiring no user intervention. Generally, agents are separated into 'cells' associated with local points and wearable computers. The experiments conducted addressed the notions of privacy, localised resource management and personalisation. It is argued that using a peer-to-peer network of wearable and ubiquitous computing components will solve problems in ubiquitous computing systems.

2.8.2 Affective computing

In Affective Computing, the emotional state of a user is monitored, usually by a wearable computer. According to psychologists, emotions such as fear, excitement, confusion and tiredness can be identified by skin conductivity, as described above in relation to the work of Picard and

Healey [90]. By monitoring these conductivity levels, the wearable can intervene in real time. As an example, in the case of tiredness, the wearable could start playing one of the user's favourite songs [164]. Augmentation of memory can also be triggered by an emotional event of confusion. Monitoring states of confusion about particular events can provide related information to the user to assist him or her in the comprehension of a particular subject.

Apart from skin conductivity, research efforts include detection of expressions of emotion, such as smile, gestures, voice strain or accelerated heart rate [164]. The system used by Picard and Healey to describe such functionality uses a bio-monitoring system equipped with sensors for respiration, skin conductivity, temperature and blood pressure and also provides an electromyogram to indicate muscular electrical activity. The prototype is based on MIT's Tin Lizzy. As stated in [164], the main problem with affective computers is the management of the variety of psychological and physiological variables. The experiments conducted indicated that a wearable computer collects long-term physiological data for an individual more effectively than traditional psychological experiments, which mainly deal with short-term data. The fact that the user wears the computer, thereby having direct contact with it, means that wearable computers are ideal for such systems.

2.8.3 Context-awareness

Context-awareness may be defined as the use of any information of the situation or state of an entity, either a person, an object or a location [66, 67]. This information can be a user's physical, informational or emotional state, a location's environmental condition (e.g. a rooms heating levels) etc. The ability to sense a state, either the user's or that of his or her environment, can be a powerful tool for a wearable since it can acquire context-related information, store it and process it in the future. Use of this information can also be real-time, as in systems that detect their own position and provide location-related information to the users [210]. Most context-aware applications involve the collection of context data by means of sensory devices.

Examples of context-aware applications are Rhodes' Remembrance Agent [187] and the 'Conference Assistant' described by Abowd, Dey and Salber [67]. The Remembrance Agent is a program which augments human memory by displaying text notes, archived emails and on-line documentation that might be relevant to what a user is typing or reading.

The Conference Assistant is essentially a wearable computer that allows interaction with other wearables at conference, helps users retrieve information related to presentations and assists note-taking. In similar systems, acquaintances can be stored as having same interests, based on the fact that their users have attended the same conference. Furthermore a camera could store images of these people along with some basic information (name, employment, area etc), which can be retrieved afterwards or added to a "contacts" list. The camera can be triggered manually or by a method of sensing the wearer's excitement about a subject, as in the Cybernetic Wearable Camera [90]. Picard and Healey state that attention and memory are correlated to a person's arousal level, which can be measured by skin conductivity. The system triggers a camera to store the images a user sees. If the user is excited about a particular subject in his or her field of view, the camera is triggered to take a picture of it.

As mentioned above, a number of different wearable computer configurations exist as far as the central unit is concerned. Equally, if not more important, are the number and types of sensory devices that can be used with a wearable computer, which may include accelerometers, GPS units and digital compasses. Most research efforts on context-aware wearable computers follow the same paradigm: a wearable computer is interfaced with a HMD and a GPS unit to provide outdoor location information [23, 167, 217] or with other sensors such as Locusts to provide indoor location data.

A somewhat different and more interesting approach was followed by Laerhoven, Aidoo, and Lowette [130]. They used a Compaq iPAQ as a central processing unit interfaced to an array of what they denote as 'cheap and simple' sensors. For that purpose they implemented a secondary sensor board based on PIC microcontrollers with a serial connection to the PDA. The board, named TEA2 (an extension to the TEA — Technology for Enabling Awareness — Esprit Project[46]), includes two photodiodes, two microphones, a dual-axis accelerometer, a touch sensor and a temperature sensor. The system uses a neural network for interpreting data. The authors argue that the use of many simple sensors may allow more accurate measurement of context-awareness although, with the current system, the use of the micro-controllers limits the number of sensors that can be used. Future work will involve increasing the number of sensors and the adaptation of the decoding algorithm to accommodate higher information load. The data were used in two applications: one logs activities such as 'sitting', 'walking', 'running' and 'bicycling' with an associated time-stamp, resulting in a daily diary. The other involves the automatic execution of applications such as an MP3 player, depending on current context.

Lee and Mase follow a similar approach [131] using their version of similar 'simple' sensors.

[46] `http://www.teco.edu/tea/`

The system comprises three main modules. The first, denoted the 'sensing module', combines an accelerometer and a digital compass, to provide data on the user's activity. The second is the 'unit motion recogniser' which classifies the movements of the user into different categories. The last module is the 'location recogniser' which uses fuzzy logic to determine the user's location. The data are used to determine the user's activity in terms of paths between pre-defined locations such as 'Printer Room', 'Library' and 'Coffee Area'.

Context awareness in general, not only in relation with wearable computers, imposes a number of problems. Abowd, Dey and Salber describe these problems and limitations in their published work on the Context Toolkit [65]. They argue that context-aware applications do not follow a unified method of interpreting data, thus making them directly dependant on the nature of the sensors used. Furthermore, this increases the complexity of implementing such applications because everything must be designed from the lowest level to be compatible with the sensory interfaces. The authors describe the Context Toolkit, a framework based on widgets, servers and interpreters, used in the aforementioned Conference Assistant. Context widgets encapsulate context information about events, locations and people. Servers, similar to widgets integrate this information in logical categories (persons, places etc). Interpreters interpret this information, providing raw data from the other components to the application in a meaningful format, such as transforming spatial coordinates to the corresponding location name. Newman *et al.* [157, 158] address this issue as well, following a more 'lightweight' approach with their information abstraction layer (IAL). The IAL provides a three-tier system between an application, a mediator and raw information from input modalities and extracts device-independent information from device specific data.

2.9 Chapter summary

It is apparent that the nature of the applications that a wearable computer can support is directly dependent on the hardware used, particularly the sensors available. The hardware used affects the user in terms of comfort, ease of use and functionality. Furthermore, problems such as inadequate displays, short battery life, weight and difficulty of interaction are of paramount importance in a wearable. Last but not least, the choice of architecture often affects the peripheral connectivity options.

Research approaches can be classified into two general categories. One involves the use of high processing-power systems — usually Pentium-class — either using the Tin Lizzy architecture or employing backpack-mounted laptops. Such systems are often used in wearable AR, explored in detail in Chapter 3. Wearable AR requires high processing capabilities, both for the CPU and the graphics processor. On the other hand, less demanding, in terms of processing power, tasks like PIM, data logging and (some) affective computing can employ much simpler wearable computers (although Picard's example [164] is a Tin Lizzy system), such as PDAs strapped to the body and interfaced to bio-monitors, accelerometers, digital compasses and 'smart badges' such as Locusts. In cases where machine vision is required, such as for indoor localisation or face recognition, the processing requirements rise and the aforementioned, more powerful, configurations generally are required.

The most popular architecture for a wearable computer is the 'Tin Lizzy' design, based on PC/104 boards. The architecture is essentially the same as the standard PC/AT (PCI/ISA/AGP based) but using miniaturised boards. However, PC/104 boards are mainly for industrial control and therefore have limited multimedia capabilities, such as 3D graphics acceleration. Furthermore, in order to implement configurations that can accommodate a large number of input modalities, more than two boards often have to be stacked, resulting in heavy and bulky implementations.

On the other hand, newer SBCs that have recently appeared have better multimedia capabilities, making them more appealing than PC/104 boards for applications such as wearable AR. Such is the 'mini-ITX' series of motherboards used in this research and described in Chapter 4. They are fully featured PCs, combining low-power consumption with relatively high processing power (both CPU and graphics processing) and do not require expansion boards. This has potential for making computers that are less bulky (i.e. flatter) than PC/104-based systems.

A number of peripherals are available commercially while some developers construct their own if there are special needs. The typical peripherals for a wearable computer include an HMD for output and a chord keyboard for input. Fully-immersive, binocular or biocular HMD designs are particularly suited for Augmented Reality applications. Monocular designs, on the other hand, are less obtrusive and therefore more useful for almost all other applications.

Input mechanisms, however, require further research. Although most researchers use chord keyboards, the author believes that this solution is far from optimum and will most probably not prove successful commercially. The use of a chord keyboard requires significant amount of practice, making them unpopular. Conversely, devices similar to the keypads used on mobile phones may be much more successful, especially given the current popularity of SMS. In some cases, audio input may also be a solution, although its use is not always appropriate as explained above.

The next chapter focuses on wearable AR implementations, examining in detail the most relevant literature on outdoor augmented reality systems and the associated challenges. The second half of the chapter focuses on wearable AR prototypes and accompanying applications.

Chapter 3

Wearable Computing and Augmented Reality

3.1 Introduction

In this chapter the author presents literature relating to outdoor augmented reality using wearable computers. The purpose of this review is to demonstrate how the problems of untethered AR (see Section 1.2) are being addressed and to examine the practices that researchers follow. The review pinpoints the need for reduced system lag, correct registration of virtual information on the real world, efficient frame rates, HMD calibration and realistic virtual world representations. These are problems encountered in all AR applications [17]. However, mobile, outdoor AR using wearable computers imposes new challenges, well beyond the inherent problems of traditional AR. These are the need for better user interfaces, the limited graphical processing power of miniaturised computers, the requirement for a high number of peripheral interfaces and the power consumption problem.

Section 3.2 describes research in outdoor augmented reality, pinpointing the aforementioned issues and the methods that developers have produced to rectify them. Section 3.3 explores AR further in the context of wearable computing, in two parts. The first deals with augmented reality tour guides, systems that are used to aid users while roaming in physical spaces such as campuses, cities and archæological areas. The second part deals with further research in wearable augmented reality in other scenarios.

Section 3.4 investigates previous research efforts in the area of software toolkits for the implementation of VE applications, focusing in particular on wearable AR. The section also reviews the most prominent solutions for creating 3D graphics applications for Linux.

Last but not least, Section 3.5 reviews usability engineering and assessment through surveys and questionnaires as encountered in the context of VR/AR and wearable computing. This aims to provide background to the assessment described in Chapter 6

3.2 Research in outdoor augmented reality

Informative descriptions of augmented reality are presented by Azuma [17, 18] and Azuma *et al.* [19], with the latter being the more up-to-date and complete. These authors explore the basic principles of AR in general, the trends and the advances achieved and describe the most prominent research. Advances in current display technologies, including HMDs, handheld and projection displays, tracking technologies, calibration and auto-calibration as well as advances in graphical user interfaces (GUI) are summarised.

They also make a series of remarks. One of the most important is that very few systems have matured beyond a lab environment, a shortcoming also stated by Brooks [45]. They attribute this lack of expansion and success on technology and interface limitations as well as social acceptance issues. They discuss the problems of registration errors, system delay, various optical issues of HMD technology and the need for high frame rates on miniaturised, embedded systems as the main obstacles in realising outdoors AR.

3.2.1 Tracking and registration

Probably the greatest problem in AR is that it is susceptible to delays and distortions relating to the registration mechanisms employed. These problems become more severe in outdoor AR where mobile, untethered systems are used. Meyer *et al.* [149] discuss the nature of orientation, motion and location trackers, classifying them in two categories: *active*-target and *passive*-target systems. Active-target systems include signal emitters (beacons) and sensors, and landmarks (fiducials) placed in a prepared and calibrated environment [255]. Passive-target systems are completely self-contained and include magnetic compasses, sensing orientation by measuring the Earth's magnetic field, inertial sensors measuring linear acceleration and angular motion (accelerometers), and vision systems sensing natural scene features, e.g. [255] (You *et al.* separate passive-target systems from inertial sensors [254] in other publications). Researchers also tend to combine trackers from these categories in order to overcome their respective shortcomings. The resulting systems are called *hybrid sensors*.

Azuma *et al.* [179] demonstrate a hybrid tracker that stabilises outdoor AR HMD displays. The hybrid tracker combines rate-gyroscopes, a magnetic compass and a tilt sensor, claiming near real-time performance. Reported peak registration errors are 2°, with typical errors under 1°, although errors allegedly become larger over time due to compass drift. The authors remark that the experimentation was done in static situations, where the user was not changing his position, and that the system is heavy and bulky.

You *et al.* [254] present a hybrid sensor for AR, combining an InterSense[1] three-degree of freedom (3DOF) orientation tracker and vision tracking technologies. The authors balance the benefits and drawbacks of using inertial sensors, vision tracking and a combination of both. They argue that inertial, passive-target systems can be robust and self-contained, yet they suffer from lack of accuracy and drift over time. Vision tracking is accurate over longer periods yet it is computationally intensive and suffers from occlusion. Their hybrid sensor combines these two technologies, complementing each other, and proves that hybrid sensors can be more accurate and less susceptible to errors.

As has already been pointed out, registration does not rely only on orientation tracking [22]. The Global positioning System (GPS), described in further detail in Chapter 5, is a satellite-based system that allows users to determine their geographical position. Unlike HMD tracking mechanisms, GPS introduces positional errors rather than angular errors. The Touring Machine [71], described in detail in Section 3.3, is an example of a system that uses GPS to determine position and a head-tracker for orientation. Höllerer *et al.* present the outdoor, subscription-based, differential GPS (described in Chapter 5) subsystem of the Touring Machine claiming an accuracy of 1–2 cm at 5 Hz, which may degrade to a metre when fewer than six satellites are visible. When standard GPS is used, accuracy can be as poor as 10–20 m. Alternative schemes may use GPS along with other location sensors; for example, Behringer [30] describes a system using a combination of GPS and a vision system that tracks horizon landmarks, claiming that it achieves more accurate registration.

Almost all the examples of sensor systems described above suffer from accuracy problems and drift over time. The ideal solution for accurate registration and localisation is a positional error of 1 mm and angular error of less than 0.5° with no drift [19]. However, accuracy requirements are application dependent; while this kind of accuracy may be needed for, say, medical applications, they are much less stringent in the research described in this thesis. The accuracy

[1] `http://www.isense.com/`

requirements for the author's achæological tour guide are discussed in Chapter 5.

The level of accuracy required by [19] is not achievable with current tracking technology [18]. If registration relies on an orientation tracker, even 1° error will result in large discrepancies as distance increases. As described in Section 1.2, in order to keep errors to a fraction of a degree, end-to-end system delay should be kept below about 10 ms, a solution that remains technically difficult and may result in expensive and complex systems. A more achievable aim for this research is that doorways in a 3D model should be sufficiently stable in terms of position that the wearer can negotiate physical doorways without difficulty [192]. Hence positional errors need to be less than about 20 cm.

Last but not least, Azuma *et al* [17] comment on the tracking mechanisms available, emphasising the need for calibration. They argue that current sensors have proven accurate only with several restrictions and that they generally require heavy and cumbersome apparatus to be carried.

3.2.2 Optical issues

The importance of the human visual channel is paramount as it is the most important for sensing the environment. Humans have an extremely sensitive vision system, able to detect small anomalies and irregularities such as mis-registrations and delays [112]. This section gives an overview of the optical issues arising from the use of HMDs and some compromises in their designs and implementations. It aims to be illustrative rather than exhaustive, as an investigation of the optical problems of HMDs is beyond the scope of this research. More extensive information is given by Kalawsky [112] and Meltzer *et al.* [148].

Human depth perception depends on many visual cues, which may be affected by an HMD's optics. Motion parallax, the relationship of an object's position, in relation to the user's field of view is probably the most important visual cue in depth perception [112]. Incorrect focus and delays in orientation trackers may affect depth perception. Stereopsis, the perceptual transformation of differences between two monocular images [112], affects depth perception and is used in binocular displays to provide stereo imagery [102]. Interposition also leads to distance awareness. If an object occludes another, then it is assumed to be closer to the user [112].

Most HMDs available commercially are designed for indoor use and have contrast ratios[2] too low for use in bright sunlight [69]. For daylight use, an HMD needs to have luminance

[2] Contrast ratio refers to the differential luminance between two adjacent points.

greater than 17 cd/m^2 [148]. Azuma *et al.* [179] report that brightness problems are found predominantly in optical HMDs, a claim also supported from independent research involving the Sony Glasstron [251]. Preliminary trials of the system described in this thesis also identified the visibility of the 3D model as being problematic, so the assessments described in Chapter 6 used custom HMD visors (see Appendix F) made from partially-crossed Polaroid[3] sheets to act as a neutral density filter in an attempt to ameliorate the problem .

When visual objects are difficult to see in comparison with the real world, the effectiveness of any augmented reality system is obviously severely compromised. However, the low contrast may be misinterpreted in terms of perspective or increased distance from the observer [69]. Moreover, if the contrast ratio varies, as it may when moving from sunlight to shadow for example, there is the possibility that these fluctuations may be misinterpreted as differential depth errors [69].

Focus can also be a problem for both video and optical see-through HMDs [19]. The effect of changing the focus of the eye for objects of varying distances is called accommodation [118]. Vergence is the effect of aiming the eye pupils towards an object by changing the angular difference between the eyes' visual axes [4]. These two effects are coupled together in natural vision. However, most optical HMDs allow a range of fixation distances while restricting accommodation to a single plane, that of the optical combiner [4]. In other words, the virtual information is projected at a fixed distance, whereas the real is not. The effect is more or less the same in video see-through models, where the camera's depth-of-view and focus settings may result in certain real objects being out of focus in the resulting composite view. The results of these discrepancies include eye discomfort [252], induced binocular stress [237], difficulty in perceiving the two images as a stereo pair [237] and incorrect perception of scene geometry [145].

A narrow field of view also imposes problems. A complete and accurate sense of space requires a very wide field of view [18, 69] as that is what humans are accustomed to. Henry and Furness [93] suggest that sizes and distances tend to appear smaller when viewed from a truncated field of view. Peripheral vision is particularly affected by an HMD and the user may not be able to see important features of the real world such as the floor, doorways and corners, making movement awkward and affecting postural control [148].

In addition, for a fixed number of pixels, resolution is inversely proportional to the field of view, resulting in an unfavourable trade-off. The greater the field of view, the lower the

[3] `http://www.polaroid.com`

resolution. Current HMD technology pixel resolutions are far less than the resolving power of the human eye [19], often leading to a difference in accommodation and therefore depth perception problems [69, 155]. Furthermore, the inter-pupillary distance (IPD) affects the convergence point of an object in a stereoscopic viewing system. If IPD is not taken into account in an HMD, the scale of the virtual image in depth may be wrong [69].

Apart from the aforementioned shortcomings, HMDs are susceptible to *binocular rivalry*, a phenomenon that occurs when different images are presented in each eye [5]. Humans use both their eyes to view the visual field, i.e. binocular vision [112], in order to triangulate distance more accurately and hence improve depth perception. Eye dominance is the priority given by humans to information from one of their eyes, suppressing the other relatively. In some case the dominant image alternates between the two eyes, resulting in binocular rivalry [112], which may occur due to differences in luminance, image complexity and scene representation.

3.2.3 HMD calibration

In order for an AR system to work, see-through HMDs must be calibrated such that the virtual models of objects match their physical counterparts. This process involves matching each object's position, orientation and size with that of their physical counterparts. By the term 'calibration' researchers usually refer to three separate processes. Pointer calibration usually refers to picking up visual markers in the environment to register the virtual image with. The actual registration is termed workspace calibration [219], which aligns the virtual world's coordinate system to the real world's coordinate system. Finally, display calibration is the estimation of the projective transformation so that objects in the virtual world appear in correct spatial positions relative to their real counterparts [21]. The effects of improper calibration are discussed by Drasic *et al.* [69].

Many research efforts attempted in the past to achieve an efficient method of see-through HMD calibration, however most have shortcomings and are considered unrealistic and tedious for practical use, especially in commercial systems, a claim supported by Tang *et al.* [219]. Azuma [21] and Bishop [20] present two methods, a static and a dynamic calibration technique. The static approach measures certain features of the HMD and the real world, such as real object position and orientation, HMD field of view, HMD centre of FOV and position and orientation offsets between the tracker and the eye. The dynamic approach uses predictive methods to achieve similar registration, with the ultimate aim of overcoming system delays. Tuceryan and Navab [231] present a display calibration method called the Single Point Active Alignment Method (SPAAM). The calibration procedure involves repeatedly aligning a single marker displayed in the HMD with a point in the real world. An extension to this method is presented by Genc *et al.* [79] for stereo displays. Like Tyceryan's and Navabd's method, the procedure involves repeatedly aligning a 3D circular disc projected through the HMD to a real circle in the workspace so that the two objects match in terms of radius and centre. Kato and Billinghurst describe a method using fiducial markers and machine vision for optical see-through HMDs [117]. Kutulakos and Vallino [127, 128] demonstrate a calibration-free system where metric information is not required as the system uses fiducial points in a rendered frame to compute the perspective projection and therefore 'place' the objects in space.

Although HMD calibration is a problem with all AR applications, the problem is very rarely encountered in the context of wearable computers. That is probably due to the fact there are few practical implementations of wearable AR that require precise registration and because wearable AR systems usually suffer from performance issues that do not allow proper investigation of HMD calibration algorithms.

3.2.4 Portability

Azuma *et al.* [19] remark that most computer systems for AR are difficult to carry when outdoors because of their weight, bulk and the number of sensors used. They add that if portability is achieved, untried applications will become feasible.

The author of this thesis believes that this particular problem is because wearable computing has not attracted enough interest from manufacturers and system integrators to implement lighter and faster systems, just as in the aforementioned case of the HMDs. Current laptop technology has improved significantly, with many systems having the fast CPUs and graphic cards required to render satisfactorily modern computer games. Most wearable prototypes use either stock laptops [71, 167], resulting in heavy and large systems, or boards from the embedded market that are not powerful enough because they are intended for industrial control mechanisms. Nonetheless, the author believes that the expansion of mobile technology and the trend for richer multimedia content on handheld devices will result in systems that are capable of rendering complex graphic scenes with satisfactory power requirements. Indicative of this trend is the expansion of nVidia's graphic processors in the mobile phone market.[4]

[4] `http://www.nvidia.com/page/handheld.html`

3.2.5 Virtual world generators and the need for high frame rates

Azuma *et al.* [19] claim that rendering is not one of the major problems in AR as registration and system lag introduce far more problems: solutions that do not use demanding 3D rendering but instead rely on image, audio or text annotations are equally susceptible to registration and delay problems. Nonetheless, rendering becomes a problem in outdoor AR that uses wearable computers and focuses on augmenting the real environment with 3D reconstructions. Unlike outdoor AR, this can be relatively easily remedied for indoor systems by the use of workstations equipped with fast graphics cards and hardware support for toolkits such as OpenGL and DirectX. Nonetheless, for outdoor systems, where portability and power consumption are important, similar system configurations result in heavy, large and power-hungry configurations. It is characteristic that the two most important wearable AR platforms that render 3D graphics, the Touring Machine [71] and Tinmith [167], reviewed in detail in the next section, are based on laptops with hardware support for 3D rendering, whereas a third solution, Archaeoguide [235], does not use detailed 3D reconstructions of archæological buildings as initially planned but instead only image-based annotations.

Ware *et al.* [240] discuss the effect that frame rate has on the correct manipulation of virtual objects. Although they focus on VR applications, their results and approach can be applied to AR. The authors define the cycle a VR system requires to update a frame in a virtual world. They argue that for reducing lag in manipulating virtual objects the frame rate must be as high as possible. Kalawsky *et al.* [115] mention that virtual environments, in general, need to run as smoothly and fast as possible to maintain the sense of presence. Richard *et al.* [191] also argue in favour of high frame rates in VR/AR systems. Human interaction with the real environment relies mainly on visual input and confirmation of distances, position, velocity acceleration and colour occurring in real-time. In a similar approach Wloka [249] argues that, if the overall lag between an external stimulus that requires rendering a new image is applied and the update of the frame is more than 300 msec, the user loses the feeling of immersion and observes misregistrations. As mentioned in Chapter 1 the overall system lag is the sum of the rendering delay, the processing delay and any delay introduced by sensory modalities [14, 182].

It should be noted that the human eye can detect low frame rates with relative ease. MacKenzie *et al.* [138] suggest that 10 frames per second (fps) are considered a minimum to achieve real-time animation. However, researchers have quite different opinions when it comes to the minimum frame rate required for a AR/VR system. Hubbold *et al.* [105], when referring to a VR system, suggest as a minimum a higher frame rate of 60 fps, arguing that less than that results in visual artifacts such as the duplication of vertical details. This is comparable to the field update rate in broadcast television and substantially higher than cinematography. They further add that a frame rate of less than 20 fps appears to be merely a fast series of separate frames. On the other hand, Billinghurst and Kato [36] believe that wearable computing tasks do not require frame rates as high as desktop users. They argue that, between a frame rate of 1 fps and 30 fps, performance on wearable computing tasks is not considerably different.

This author believes that frame rate is one of the most important bottlenecks in wearable AR systems that use 3D graphics, where the hardware requirements enforce a compromise between weight, power consumption and processing power, a claim supported by Piekarski *et al.* [169] and Baillot *et al.* [24]. Low power consumption normally goes hand in hand with low graphical performance, and hardware support for 3D rendering tends to be the first capability that is discarded when designing low-power graphics processors.

The author of this thesis, agreeing with MacKenzie *et al.* [138] supports that a frame rate of 10–15 fps, slightly faster than that of old mechanical cameras, is an acceptable minimum for wearable computing prototypes as the lag from the sensory modalities, especially GPS, is at least as problematic. However, in an ideal system, where sensors introduce lag in the order of a few milliseconds, the frame rates would need to be high to maintain realism and the feeling of presence [202].

The type of virtual information rendered has a direct effect on overall system performance. Most examples of outdoor wearable AR reviewed in the next section use a limited set of virtual objects in the form of virtual tags or simple virtual worlds that are not computationally intensive in order to introduce the smallest possible rendering delay.

3.3 Wearable computers and augmented reality

3.3.1 Augmented reality tour applications

Touring Machine and MARS

In 1997 Feiner *et al.* pioneered research in the field of wearable augmented reality with Columbia University's *Touring Machine* [71], which has evolved into the *Situated Documentaries* [100] and the MARS [98] systems. These employ a GPS-equipped wearable computer to provide hypermedia presentations that are integrated with the actual outdoor locations to which they pertain.

The prototype used a tracked, see-through I-glasses HMD, subsequently replaced by a Sony Glasstron HMD and a hand-held pen computer to present 3D graphics, imagery and sound superimposed on the real world. A backpack wearable computer based on a Fieldworks 7600 industrial portable computer with a 3D graphics card supporting OpenGL served as the main unit (fig. 3.1). A Mitsubishi Amity hand-held computer was used for output and stylus-based input as mentioned in Chapter 2. Orientation tracking was done with a magnetometer mounted on the HMD and positional tracking with an Ashtech Differential GPS system.

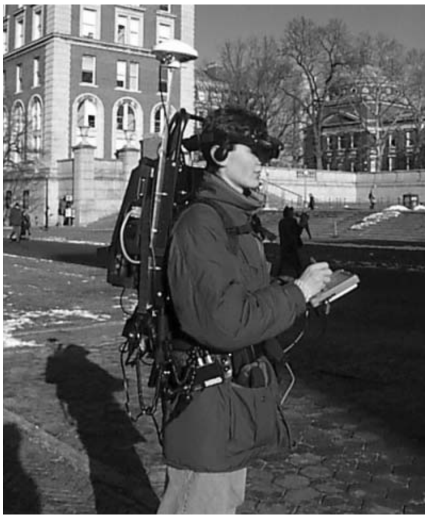


Figure 3.1: The Touring Machine from Columbia University

The user roams around the university campus and is able to see information in the form of virtual tags, indicators and point-of-interest flags through the HMD and the handheld computer. The content is arranged in three types: audio is presented through the HMD earphones in the form of narration or audio extracts; some images are presented through the HMD in the form of world tags, others in the form of 3D synthetic environmental replacements, such as buildings; and finally information is also presented through a normal web browser on the handheld computer in the form of video or images.

The authors state that, during testing, people that used the system found the wearable computer heavy (it weighs a little less than 20 kg). The weight is understandable as the authors opted for power at the expense of comfort. Feiner *et al.* state that a graphics card that can support 3D graphics acceleration and OpenGL, in order to render high-quality 3D information, is required in an AR system. The lack of such functionality in embedded systems, single board computers or even laptops in 1997 led to the use of the Fireworks 7600 which, along with batteries, weighs around 19 kg.

GUIDE

A significantly different approach to mobile tour guide systems was demonstrated by Lancaster University's GUIDE [51]. This provides city visitors with location-related information using a tablet PC equipped with a radio-LAN (802.11b) interface. GUIDE does not attempt to produce 3D reconstructions of the surroundings but instead provides text and images related to the user's position. A number of base stations around the city provide each user with information relevant to their position.

The authors argue that GPS, which they used in early versions of the system, does not provide any advantages in such an environment when compared to network-based location mechanisms. In particular, GPS requires at least four satellites to be in view in order to obtain even a moderately accurate position fix, yet this is rarely possible in the "urban canyons" formed by tall buildings.

Smart Site

A similar system to the Touring Machine, but developed at Carnegie-Mellon University, is known as Smart Site [253]. This utilises a wearable computer and a laptop to provide an intelligent tourist assistant. The system architecture caters for multi-modal input and output. Location information is obtained from a GPS unit, while a camera provides visual input. A microphone and headphones provide audio input and output. Speech and gesture inputs may be used. The authors argue that for a tourist system it is important to use a number of input modalities to accommodate different scenarios. A user can roam around a site, derive location information from the GPS unit but also request for further details by speech or by gesture.

Tinmith-Metro

A newer and equally significant system in this area is Tinmith-Metro, an interactive augmented reality 3D constructive solid geometry modeller [167, 169, 170, 171], based around a wearable computer with machine vision capabilities via a USB camera. User input is principally via a set of pinch gloves, allowing the user to execute commands using menus linked to finger gestures; indeed, the authors argue that desktop user interfaces should be avoided and replaced with speech recognition, camera input and hand-gestures. Position is determined using differential GPS, while a tracker measures head orientation. The system is capable of generating 3-D models of the external surfaces of buildings as the user roams, and can place pre-fabricated 3D

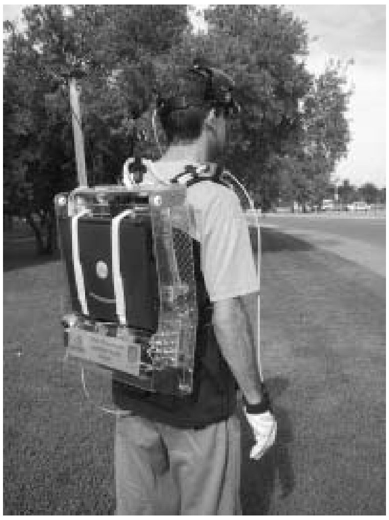


Figure 3.2: The Tinmith-4 Wearable from University of South Australia

objects within a scene. Objects are created by fitting infinite planes to surfaces and using their intersections to define buildings.

Tinmith-Metro is based on a Gateway Solo laptop with an ATI Rage 3D graphics card, carried in a hiking backpack (fig. 3.2). A Sony Glasstron PLM-700e monocular display is used for output and an Intersense IS-300 tracker provides orientation information. Positional information is derived from a Trimble[5] Ag132 GPS unit, using free or subscription-based differential correction services to calculate positions in real-time with claimed accuracies of under 50 cm. The Tinmith configuration is actually quite similar to Columbia University's MARS system, the main difference being the input mechanisms. The gloves used in Tinmith are effectively a pair of gardening gloves, with special conductive tape attached. A microcontroller detects gestures by polling the gloves 30 times per second and sends the information via a serial link to the laptop.

Tinmith-Metro enables users to control a 3D constructive solid geometry modeller [167] for creating graphical representations of large physical elements such as buildings. These elements are manipulated in real time using an novel user interface, "Tinmith-Hand", using the pinch gloves and hand tracking. A USB camera tracks fiducial markers on the pinch glove thumbs which manipulate the cursor. Menu-option selections are done with the rest of the fingers by pressing them against the thumb, with each finger mapped to a menu option. Tinmith-Hand controls the graphics modeller, a custom-made rendering system similar to SGI's Inventor. The authors imply the need for a hierarchical scene graph (see Chapter 5) in order to render de-

[5] `http://www.trimble.com/`

manding 3D graphics. Their initial system of 2001 rendered at 2 frames per second.[6] In 2003 the system was updated to the aforementioned configuration, yielding a claimed speed of 40–50 frames per second, manipulating simple objects such as a virtual table.

This author considers this effort as the most complete and up-to-date effort for augmented reality. In terms of performance it claims improved accuracy and rendering speed compared to older implementations. Nonetheless, the system is again heavy, weighing approximately 16 kg. There is a fair amount of cabling associated with the apparatus and the battery life is similar to modern laptops. Furthermore, the type of objects it has to render are fairly simple, not realistic or complex. The authors emphasise that there was no effort on their behalf to miniaturise the system with custom components and that their main concern was that the overall system architecture should prove the concept of AR.

Archeoguide

Archeoguide [235] was an EU-funded programme that aimed to produce a virtual tourist guide. Archeoguide offered an augmented reality tour of the ancient site of Olympia in Greece, based on wearable and mobile computers and wireless networking technology. The system intended to allow the user to participate in personalised tours while roaming the site and provided audio-visual information on the ruins in the forms of AR monument reconstructions and audio narration.

One of the mobile systems used in Archeoguide was a laptop-based wearable computer. The initial prototype featured a Pentium III 800 MHz CPU with 256 MB of RAM, a 20 GB hard disk and an nVidia graphics card. A Garmin[7] 35-HVS GPS receiver and a Precision Navigation[8] digital compass provided position and orientation information respectively. A pair of Sony Glasstron glasses provided output. A helmet-mounted Philips[9] PCVC 690 K USB camera was used for visual tracking.

Its architecture, although similar in visual output to the one that forms the focus of this author's research, has some fundamental differences. A central server holds information related to the site in the forms of images, text or audio, including an image database, and is linked to the wearable computers via a 802.11 wireless LAN. Information is transmitted to the mobile

[6] http://www.tinmith.net/demos.htm

[7] http://www.garmin.com/

[8] http://www.precisionnav.com/

[9] http://www.philips.com/

computers according to the user's profile, position and orientation.

Archeoguide uses a hybrid technique for position determination. GPS provides coarse positioning. Once the user is in front of one of the buildings to be augmented, standing in a pre-defined position, a computer vision algorithm tracks the remains in front of the user and places accordingly an image of the building, downloaded from the central server. These images are views from pre-defined viewpoints of the ancient buildings, made in Discreet's[10] 3DSMax and are essentially images, not 3D models.

Although Archeoguide's results seem impressive in terms of the detail and quality of the information augmented, the system, in this author's opinion, does not constitute real-time augmented reality. The use of pre-rendered images may simplify rendering requirements but it does little for the flexibility of the system and does not allow for real-time, always present augmentation. If a user is not standing at a predefined position, no synthetic information is presented. Furthermore the use of images, viewed in specific locations rather than complete 3D graphical models, does not allow the user truly to roam in the synthetic environment. The author also believes that the client-server approach used does not scale properly to a large number of users. If more than one wearable user roams around the archeological site, different imagery has to be transmitted to each user. This imposes an significant overhead on the server when a large number of users are involved and the bandwidth of 802.11 is likely to be a bottleneck of the system.

On the other hand, optical tracking, as in the case of Tinmith-Metro, is again used to enhance the registration of virtual information on top of the real ruins and offers the potential for an increased level of accuracy. GPS can only achieve an accuracy of 2 m today (see Chapter 5) and even that requires a sufficient number of satellites to be viewable. Optical tracking is one of the methods used by researchers to increase positional accuracy and improve scene registration.

3.3.2 Other wearable AR research applications

EyeTap

Aimone *et al.* [3] use their EyeTap vision tracking system to investigate optical tracking and to correct image registration, arguing that vision-based systems can achieve correct registration with minimal drift. The system uses the VideoOrbits [2, 143] vision-based tracker, for low-frame-rate drift compensation and two gyroscopic elements for high-frame-rate compensation and

[10] `http://www.discreet.com/`

enhanced accuracy, resulting in a hybrid "Gyroscopic/VideoOrbits" System (GVS). Its developers claim that the system can register images to the real environment with less than 2 pixels error at 12 frames/s. In theory, the system's functionality is independent of the environment of the user but it apparently performs better in well-lit, static environments. Furthermore, the authors emphasise the low cost of the system.

Fung *et al.* [77] further investigate the acceleration of 3D graphics achieved using the EyeTap system. The authors use OpenGL (described in Section 3.4) to evaluate the performance of their algorithm in registering images to the real environment. Their investigation is done using three methods. Initially, only software rendering is used. Direct rendering (i.e., hardware acceleration) is enabled in the second approach. The third approach involves the use of the VideoOrbits algorithm, using different graphics cards as evaluation platforms, yielding reasonable results. These showed that hardware-accelerated rendering is significantly faster than software only — hardly a surprise. The authors conclude that OpenGL on hardware-accelerated systems can significantly improve the VideoOrbits algorithm and, indeed, any repetitive image registration algorithm.

Hand-held AR

A different approach to augmented reality is explored by Newman *et al.* [156] in the BatPortal project. A hand-held PDA is used as the presentation medium, connected to a wireless beacon called a 'Bat' which provides location information. The user wears a second Bat, resulting in a vector that specifies the user's direction of view. Both Bats are also tracked in relation to the environment. The location and viewpoint information is transmitted to a workstation which processes it. The PDA, an HP iPAQ, acts as a simple output device. This approach differs from those described previously in the use of a PDA, yet the overall architecture is similar to that of Archeoguide. The implementation, based on a centralised system, may be problematic when the number of the users increases significantly, although the authors claim that their system is successfully used by more than 50 participants during everyday tasks in an office environment.

Wagner et al. [236] also use a hand-held computer equipped with a commercial camera that tracks optical markers in real time. The authors argue that most systems following the laptop-in-a-backpack paradigm are heavy, difficult to master and socially unacceptable. Furthermore, the authors do a careful analysis of potential architectures for AR applications. A main wearable unit that does all the processing required for AR and a thin client approach are considered two extremes of the spectrum of potential configurations. The authors fragment an AR application into five stages: visual acquisition, tracking, scene processing, rendering and display. Any intense computational stage is ported to a central server while the rest remain on the mobile unit. This, according to the authors, minimises the dependencies between the roaming main unit and the central server.

Wagner *et al.* describe an application where all the tasks are either performed on an iPAQ PDA, or the tracking only is passed to a central workstation. Switching between these two strategies is done dynamically and seamlessly. The PDA is equipped with a Wireless LAN interface and a camera connected to the CompactFlash slot. Particular attention is given to the 3D graphics performance of the system. The authors pinpoint the need for fast floating-point calculations, not available on their PDA, for the geometry part of the scene generation. Furthermore, the lack of 3D acceleration imposes further problems on lighting and texture rendering. The authors implemented their own software renderer, called SoftGL, which uses a subset of OpenGL (see Section 3.4). Their approach to AR is relatively lightweight, using simple overlays and markers.

This author's opinion is that their approach is practical and realistic when it comes to AR applications on low-processing-power platforms. Fragmenting an application into stages and allocating them to a more powerful server should decrease the workload of such platforms. Nonetheless, care must be taken when the system accommodates large numbers of users as the number of central servers and the available communication bandwidth imposes a trade-off. Once again, the need for hardware acceleration is noted as the complexity of the augmented graphics increase.

3.4 VR and AR software

3.4.1 Hardware abstraction

Virtual and augmented reality applications often use tracking devices to derive orientation and location information. Tracker-abstraction layers are often employed to provide an interface method between different types of tracking sensors, so that multimodal interfaces can be used with one application [157, 165].

Examples of such hardware abstraction toolkits are the MR Toolkit, presented by Shaw *et al.* [201] and VRPN, presented by Taylor *et al.* [220]. Mainly oriented towards shared virtual environments systems, both MR Toolkit and VRPN allow transmission of sensory data over a network. The use of such hardware abstraction layers allows the replacement of one sensor with another without requiring re-coding of applications.

An alternative approach includes systems that provide an inner kernel [165] that connects together various components, such as sensor sub-systems and data processing functions, to rendering toolkits such as OpenGL[11], OpenGL Performer[12], and Open Inventor[13], reviewed in the next sections. Such systems are VR Juggler, presented by Bierbaum *et al.* [33], MAVERIC, presented by Hubbold *et al.* [103] and DIVERSE, presented by Kelso *et al.* [119]. These systems are similar to MR Toolkit and VRPN as they allow sensor data to be transmitted over a network. Alternative sensory schemes can be used without requiring the modification of applications.

Another approach is implemented in OpenTracker, by Reitmayr and Schmalstieg [185]. The system, based on the use of XML configuration files, implements hardware abstraction as with the aforementioned toolkits by using filters. The system uses a set of three types of objects: one to read values from sensors, a second to filter them and process them, and a third to output them.

3.4.2 Distributed systems

A popular approach to VR systems is the use of distributed architectures. Most research on such systems focuses on the communication protocols and not on the toolkits themselves. Examples include SIMNET, presented by Calvin *et al.* [50] and NPSNET, presented by Macedonia *et al.* [136]. These examples transmit over a network sensory information and not scene-graphs (see Section 5.8) or pre-rendered images. In order to allow greater flexibility with the underlying network protocols, a modular solution was proposed by Watsen and Zyda with the Bamboo system[241]. Bamboo uses a kernel that loads secondary modules, when the information that arrives from the network cannot be interpreted. Once the secondary module loads, Bamboo uses it to process incoming information. This allows run-time reconfiguration and does not require the local storage of modules.

A more applicable example in the context of this thesis is the aforementioned BARS system [134]. The system distributes event data to wearable computers, excluding geometry data and pre-rendered images. The transmitted data are used to pass on information to mobile, wearable AR users and update locations and context-related information in a database. Bauer *et*

[11] `http://www.opengl.org/`

[12] `http://www.sgi.com/products/software/performer/`

[13] `http://oss.sgi.com/projects/inventor/`

al. [29] also present a component-based framework named DWARF, mainly addressed to wearable systems and intelligent environments. The framework uses a layered services architecture, allowing application developers to focus on the implementation of the application without worrying about the implementation of services or the fact that they are using a distributed system [29].

Some systems provide more complete distribution, not only small parts like the ones already mentioned, where entire applications and scene graphs are synchronised between multiple hosts [165]. Of particular importance to the research effort described in this thesis is the COTERIE testbed developed by MacIntyre and Feiner [137], designed to assist the implementation of distributed virtual environment applications and also used in the Touring Machine wearable system [71]. The testbed uses Obliq-3D[14], a high-level 3D animation system, written in the Pascal-based Modula-3 programming language[15]. COTERIE was developed in order to provide a more complete solution of hardware abstraction than the aforementioned systems, with main feature being a distributed shared memory. Integrated into the system are components that support threaded processing, sensor abstractions and 3D rendering.

The Studierstube framework, presented by Schmalstieg *et al.*, is also used for the implementation of AR applications [195]. Studierstube is a set of C++ classes built on top of the Open Inventor toolkit and is interfaced to the aforementioned OpenTracker [185]. The Open Inventor toolkit was subsequently extended by Hesina *et al.* [94] to a distributed version that transmits scene-graph changes over a network. Contrary to MacIntyre and Feiner's system, where all applications are implemented using shared memory, Studierstube follows the opposite path of implementing applications inside a distributed scene graph [165].

Other examples of distributed systems used in virtual environments are DIVE, developed by Frecon and Stenius [75], and the Avocado framework developed by Tramberend [229], similar to Hesina's Distributed Inventor [94] and MacIntyre's distributed scene-graph from COTERIE, named Repo-3D [137]. DIVE addresses the problem of virtual environment scalability by making extensive use of multicast techniques over a peer-to-peer network as opposed to a client-server model, and by partitioning the virtual universe into smaller regions. These regions can be requested or disregarded based on custom semantics [75]. Avocado is based on OpenGL Performer and is therefore addressed to applications with complex virtual worlds. It uses a scripting language to allow run-time modifications.

[14] `http://www.research.compaq.com/SRC/3D-animate/home.html`

[15] `http://www.research.compaq.com/SRC/modula-3/html/home.html`

3.4.3 Technologies for building 3D models under Linux

There are a number of industry-wide standards for developing 3D graphics applications. One of the most widely used is the Virtual Reality Modelling Language (VRML). VRML is a 3D navigation specification (ISO14772) which enables the creation of interactive 3D web content, essentially being a specification language for scene-graphs. VRML uses a viewer, a discrete application that interprets `.wrl` files and renders the scene appropriately. Nonetheless, using alternative input mechanisms to the keyboard or the mouse, such as the GPS and HMD modalities, requires either a custom viewer or modifications to existing ones, a solution that the author believes is tedious and inefficient. Various viewers, such as FreeWRL[16] for Linux and Cortona[17] for Windows, are available; however none allows alternative input to a keyboard or a mouse without requiring major alterations.

An even more commonly used development toolkit is OpenGL, developed by Silicon Graphics[18]. This is a platform-independent graphics API, the *de facto* interface supported by modern graphics cards, some of which have particular optimisations to increase rendering speed and incorporate special features (dynamic lightning, textures etc.). To provide further functionality for application development, Silicon Graphics developed Open Inventor [214] and OpenGL Performer[19] [194]. As has been previously mentioned, Performer was designed for high-performance visual simulation applications whereas Inventor is focused on more complex user interfaces.

Contrary to VRML, OpenGL allows complete control of the input modalities that are to be used. By specifying the position of the user and the point he or she is looking at, a vector representing the line of sight is formed. Using modalities such as GPS and HMD-tracker this vector can be rotated and moved appropriately in real time, changing the user's view of the model to mimic the user's change of view in the real world. Due to these features, OpenGL is the toolkit used in the implementation of this application.

3.4.4 OpenGL, GLU and GLUT

OpenGL is a platform-independent software interface to graphics hardware [203] that provides a powerful, yet primitive, rendering command set. It does not include commands for performing

[16] `http://freewrl.sourceforge.net/`

[17] `http://www.parallelgraphics.com/products/cortona/`

[18] `http://www.opengl.org/`

[19] Originally named IRIS Performer and Inventor

windowing tasks or obtaining user input; instead, it uses whatever windowing system is used with the hardware platform [203]. Likewise, OpenGL does not provide high-level commands for modelling three-dimensional objects; these have to be built up from a set of geometric primitives such as points, lines, and polygons. A number of libraries exist as extensions to the OpenGL core that simplify the implementation of OpenGL applications; those used in the tour guide application are GLU and GLUT.

The OpenGL Utility Library (GLU), a standard part of every OpenGL implementation, provides modelling features such as quadric surfaces, Non-Uniform Rational B-Splines (NURBS) curves and surfaces and other utility functions [203]. GLU also provides support for matrix manipulation and coordinate projection (transformations), polygon tessellation and error handling[20].

The OpenGL Utility Toolkit (GLUT) is a window-system-independent toolkit, with ANSI C and FORTRAN bindings, for writing OpenGL programs[21], written by Mark Kilgard. It implements a simple windowing API for OpenGL. GLUT includes several routines dealing with window creation, a callback-driven event loop, support for various pre-defined objects, font support as well as support for user input. The most important features used in the tour guide application (Chapter 5) are the OpenGL viewing mechanism, the GLU `gluLookAt()` function and the GLUT callback event loop.

3.5 User assessment in VR/AR and wearable computing

One of the most important aspects of the design process of advanced human interaction systems such as wearable AR prototypes is usability engineering [44]. The International Standards Organization (ISO) defines usability [109] as:

> The effectiveness, efficiency, and satisfaction with which specified users achieve specified goals in particular environments.

Usability engineering, as defined by Tyldesley [232], is a process which specifies the usability of a system quantitatively. As pinpointed by Kalawsky [113], usability engineering involves both scientific and engineering processes, covering human-factors (HF) based evaluations. Human factors engineering is a multi-disciplinary science, investigating users' interactions with systems.

[20] Specification available from: `http://www.opengl.org/documentation/specs/glu/`

[21] Specification available from: `http://www.opengl.org/documentation/specs/glut/`

Its ultimate goal is to aid the design of systems, operations and environments so they are compatible with human capabilities and limitations.

By controlling the usability engineering process, Good *et al.* [81] suggest that it is possible to define usability goals through metrics, set planned levels of usability, analyse the impact of possible design solutions, incorporate user-derived feedback, and iterate through the 'design–evaluate–design' loop until planned levels are achieved. Kalawsky [113] adds that questions related to human factors often are dimensionless and difficult to relate to more objective terms. Hence there are no generally-accepted methods of measuring human performance, an important aspect of HF most commonly related to understanding human interaction with computer systems such as AR/VR systems.

In the context of wearable computing and in particular wearable AR, very little practical assessment of prototype systems has been attempted. Previous wearable AR systems for 3D architectural reconstructions have not been evaluated in real operating conditions, (i.e. *in situ*) in terms of their usability. On the other hand, Lyons and Starner [135] remark on the need to have user feedback to aid more common quantitative assessment methods.

However, a number of user assessments in VR/AR and wearable computing have been carried out, pinpointing the need for practical evaluations of the prototypes in terms of human factors engineering. Such assessments are often based on questionnaires which are used to produce attitude metrics; users' opinions are important in usability engineering [175].

One of the most popular and extensively researched methods among VE/VR researchers is the Simulator Sickness Questionnaire (SSQ) [120, 121]. The SSQ consists of a checklist of 16 symptoms arranged in three groups concerning nausea, oculomotor symptoms and disorientation and contains a 4-point answering scale for all items. A weighted scoring algorithm results in a score reflecting the overall discomfort level, known as the total severity (TS) score.

Another important tool for user performance evaluation is VRUSE [113, 114, 115] assessing the usability of virtual reality (VR) systems. VRUSE is based on a questionnaire, divided into ten sections: functionality, user input, system output, user guidance and help, consistency, flexibility, simulation fidelity, error correction/handling robustness, sense of immersion, and overall system usability: it comprises some 100 questions, derived from a larger set of 250. The resulting data can be used to diagnose problems in the design of VR interfaces and systems.

Pausch *et al.* [160] present the results of a user study exploring the performance of users in a comparison of HMDs and stationary displays using a post-experiment questionnaire. Following a less structured approach, Bowman and Hodges [39] investigate the manipulation of virtual objects in immersive VEs through informal user interviews. In addition, Bowman and Wingrave [40] present the design of TULIP, a menu system using pinch gloves and compare them to pen and tablet menus. A series of questionnaires collect data on users' perceived level of comfort and preference of use. Ames *et al.* [8] perform a questionnaire-based examination of non-ocular and ocular symptoms of using HMDs in VR environments. They argue that, when examining symptoms of HMD use, post-experiment questioning needs to be done within a small time interval (<6 minutes) as most symptoms fade after that time.

The sense of presence is also assessed by means of questionnaire-based evaluations. Witmer and Singer [248] describe their presence questionnaire (PQ), which measures presence in VEs, and the immersive tendencies questionnaire (ITQ) that measures differences in the tendencies of individuals to experience presence. However, their approach is openly challenged by Slater [208]. Prothero *et al.* [178] describe a questionnaire investigating the relation between vection (visually-induced false sensation of self-movement) and presence. VRSAT [113] is a questionnaire-based programme aiming to develop a reliable and robust method of obtaining situational awareness measures. It attempts to investigate various factors that contribute to the sense of presence.

Siegel and Bauer [205] present their user assessment methodology in the context of wearable computing. Their approach uses three questionnaires, administered before, during and after a field usability evaluation of a maintenance support wearable computer at a US Air Force Reserve Tactical Fighter Squadron. In addition, they use video recordings of users' comments during the evaluation and post-experiment interviews. The purpose of the assessment was to evaluate the use of the system's input dial, HMD and application software. Likewise, Billinghurst *et al.* [34] investigate wearable information displays via a user evaluation experiment and a post-experiment questionnaire. Suomela *et al.* [217] follow a different approach by getting direct feedback from users while performing certain tasks. Their system consists of a wearable computer with a wireless connection to a second computer, used for real-time control and monitoring. Suomela *et al.* claim that the system performed well during user evaluations and that it allows a test supervisor to monitor the experiment and record sensory data, for post-processing.

Most examples above attempt to collect user feedback on the usability of the system by means of questionnaire and in conditions resembling as much as possible normal operating conditions. A similar approach is followed in this research effort, described in Chapter 6.

3.6 Chapter summary

Most of the research efforts described in the first two sections of this chapter suffer from a series of problems. In cases where the system is powerful enough and fast enough to render 3D graphics using hardware acceleration, the prototypes produced are bulky and heavy. The interfaces used also suffer from problems, resulting in systems that cannot evolve into commercial systems and merely prove the principle of mobile AR. In systems that are lightweight, such as the hand-held AR examples provided, the level of augmentation is minimal. Furthermore, almost all of the systems suffer in their location/orientation sensing and consequently in registration of virtual information upon the real environment. The use of hybrid sensors improves accuracies but the prototypes produced are also heavy and cumbersome.

Due to these constraints, such systems are unlikely to become commercial solutions and fail to make mobile outdoor augmented reality an appealing experience. Researchers are able to understand the potential of wearable AR, yet most people, including the author, who have seen or used such systems are sceptical as to their effectiveness and practicality. In that respect, user assessment is a powerful tool in the future development of advanced human interaction systems, including wearable AR prototypes. The latter have to become more practical in order to be evaluated by users under normal operating conditions and not in simulated environments.

The research presented in this thesis aims to improve the practicality of mobile augmented reality. Critical to this is obviously the hardware involved, particularly its support for the appropriate range of sensors and its ability to perform rendering with hardware acceleration. However, this must be achieved without compromising either battery life too much and with a rather lightweight solution. The latter criterion in particular mitigates against laptop-based wearables as they are invariably too heavy. Instead, drawing inspiration from the early 'Tin Lizzy' designs, the author has designed and constructed a hardware platform centred around a mini-ITX single-board computer; it is believed to be the first wearable that uses this board.

The following two chapters describe the design and implementation of the wearable prototype, called Romulus[22] (Chapter 4), and the Gosbecks tour guide (Chapter 5). Design requirements have been divided in to those related to hardware and those related to software, and are presented separately in each chapter to aid their analysis and discussion.

[22] The choice of names originates from the first wearable of the Vision and Synthetic Environments Laboratory, named Rome [157].

Chapter 4

The Romulus Wearable Computer

4.1 Problem statement

Chapter 3 reviewed prototype systems developed at several institutions for proof-of-concept experiments in mobile AR. Due to the requirement for good 3D graphical performance inherent in AR [24], all the systems reviewed were heavy, cumbersome and laptop-based. The author contends that significant research is required to convert these proof-of-concept systems into something that is small and light enough — and easy enough to use — for (say) a grandmother visiting an archæological site. He further contends that the value of any assessments of the system in *in situ* experiments is much reduced if the wearer has to remain uncharacteristically static, avoid bending down, and so on; assessment needs to be done under normal operating conditions and not in simulated environments [135, 192, 193].

This chapter presents the wearable computer platform used in this research, designed to address the above issues. Section 4.2 provides a justification for the use of commercial-off-the-shelf components, discussing briefly the implications of this approach. Section 4.3 presents the design requirements for the wearable computer. Section 4.4 discusses in detail these requirements, providing insight to the proposed design. Section 4.5 describes the final design of the wearable computer and Section 4.6 describes its construction.

4.2 The use of COTS

One aim in designing a custom wearable computer for mobile AR is to minimise cost. This, and the wish to facilitate easy replication of the hardware, together suggested the use of commercial-off-the-shelf components (COTS) when possible.

Literature on studies of the use of off-the shelf components components is limited and mostly refers to software [38, 46, 47]; however the conclusions drawn are applicable to some extent for hardware too. COTS are available in multiple identical copies, allowing duplication and rapid development and are encountered in a number of different configurations and with processing power capabilities. Researchers who do not have access to manufacturing resources can assemble prototypes with relative ease using COTS components, an approach widely adopted in the wearable computing research community. In many cases the cost of implementation and maintenance is much lower than with fully-custom systems.

On the other hand, Hall and Naff [87] argue that one of the most important constraints of COTS-based designs is the availability of the required components to satisfy the target systems requirements. Bearing in mind that the marketplace, and not the individual requirements of a particular system, drive the availability of COTS components [47], this can be challenging. Even when such components exist, implementation cost and maintenance factors also need to be considered. They further suggest that mixed approaches, using carefully-selected COTS components along with modules developed from scratch, may result in lower cost and shorter development time. Providing an example of a system, the Integrated Processor, [87] states that cost and development time was reduced by building only the chassis, backplane, I/O expansion cards and power supply and relying on COTS for the rest of the components (SBC, display processor, memory and operating system). Interoperability between components can also be a problem [38], partially solved by the use of standards — like the use of PC/AT-compatible hardware in this case.

A similar approach was followed in the implementation of the Romulus wearable computer, the hardware platform used for the wearable AR research described in this thesis. The development of this system depended on the availability of capable platforms, in terms of processing power and in particular 3D graphics hardware acceleration, supported by the operating system used. Cost, size, weight and the I/O interface capabilities of the main platform were also important factors, as will be described below.

4.3 Requirements and conceptual design

This section presents a summary of the hardware requirements, analysed in detail in Section 4.4. These requirements are:

Requirement 1: The hardware to be designed, implemented and used in the experiments in wearable AR needs to be a relatively light (<5 kg) wearable computer, equipped with a colour HMD for output and location and orientation sensors (fig. 4.1). Current wearable AR prototypes that employ 3D reconstructions are in excess of 14 kg. [71, 167].

Requirement 2: The wearable is required to be powerful enough to render a 3D reconstruction of a Roman temple containing approximately 30,000 polygons with a frame rate of at least 10 fps. This frame rate is considered an acceptable minimum, bearing in mind the portable, low-power nature of a wearable computer. The wearable therefore has to be built on a platform with a relatively powerful graphics processor and with hardware accelerated drivers.

Requirement 3: The wearable has to use the Virtual i/O I-glasses HMD and a Garmin Handheld 12XL GPS unit.[1] These components are available in the author's laboratory from previous research [111, 153, 157]. The use of these components significantly reduces the cost of development of this platform and allows duplication of the prototype, as these units are available in the laboratory in numbers.

Requirement 4: The platform must have an adequate number of external interfaces to support the connection of the location and orientation sensors and a wireless Ethernet adapter for connectivity.

Requirement 5: A storage medium is required for the operating system and the application software to be developed.

Requirement 6: A protective enclosure which is light, as compact as possible and has good RFI/EMI shielding properties should encase the main components.

Requirement 7: Operational life has to be at least two hours under normal processing load.

Requirement 8: Overall cost for the main unit should be kept under £300.

A conceptual design using UML[2] notation [31, 32], depicting the main system components is show in figure 4.1.

[1] These components are further discussed in the next chapter

[2] `http://www.uml.org/`

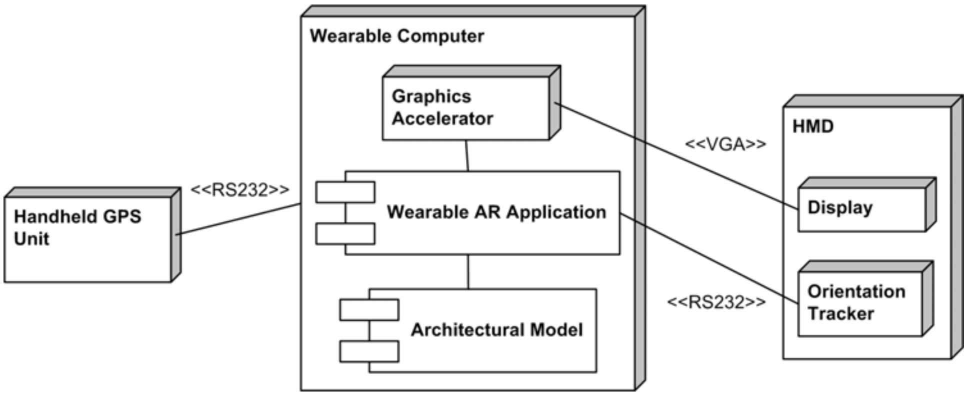


Figure 4.1: Conceptual UML deployment diagram

4.4 Hardware requirements analysis

This section analyses the above requirements in detail, describing their implications. Issues of functionality, performance, ergonomics and construction and maintenance are discussed.

4.4.1 Functional and performance

As was described in Chapter 2, Linux was chosen instead of Microsoft Windows due to its ease of customisation, its independence from graphical user interface and the availability of open source APIs for the development of 3D graphics. Due to the requirement for fast rendering of the 3D model, hardware-accelerated software drivers for Linux were required. Rechargeable NiMH batteries would provide power to the system. Such batteries are available in adequate capacities and have low cost. Memory and storage requirements were not strict as memory and storage modules are easily and cheaply purchased. Wireless connectivity was also required, for experimentation with the homegrown GPS solution described in Chapter 5. A wireless USB dongle or a PCMCIA interface with a wireless LAN card would be sufficient. Wired ethernet connectivity was essential for development purposes and system software upgrades.

4.4.2 Human-Factors engineering (Ergonomics)

The weight of the wearable ought to be less than 5 kg, including interfaces, location and orientation sensors and batteries. The weight of the individual components should be evenly distributed around the wearer's body. The choice of a vest-based wearable, like SmartVest [197] and MIThrill [234], was preferred to the common alternatives, of a backpack or belt-mounted wearable [124]. Cabling can to be concealed in the interior of the vest, trimmed to size to aid comfort and special lockable connectors can be used avoid accidental disconnections. Although a single prototype was to be developed, several vest sizes may be required in future versions.

4.4.3 Construction and maintenance

The system ought to be simple and cheap to build, to aid construction and replication. Single-board solutions, compatible with PC/AT are preferred. Systems can be easily built and upgraded using such motherboards along with spare components from the PC market and do not demand any additional hardware experience other than that required for building desktop PCs.

The main processing unit of the system required to be encased in an aluminum enclosure. Aluminum is light and strong, has good RFI/EMI shielding characteristics, is easily shaped in fine detail with inexpensive tooling and allows greater heat dissipation than steel or plastic [9, 61]. Ventilation holes and PC fans were to be used only if was deemed necessary.

4.5 Final design and development Timeline

This section describes the final design of the Romulus wearable, briefly discussing the timeline followed. The first sub-section describes an early effort from the author to construct a basic wearable computer, named Remus, in order to aid early development of the application, described in Chapter 5. The following sub-sections focus on Romulus.

4.5.1 The Remus test-bed prototype

For this research effort, two hardware platforms were actually implemented. The first, Remus, was a Tin Lizzy system used for early practical assessment and code development and is described in more detail in Appendix B. It does not constitute a novel system but only a test-bed for the early development stages of the software described in Chapter 5. It was built during the first year of the author's research and was used for 6 months for assessment of the GPS and HMD tracking software. Remus was used at the Gosbecks site with early versions of the AR tour guide software discussed in Chapter 5, aiding informal performance evaluations of the complete system.

The reader should note that the system does not follow strictly the requirements set described above; instead it was designed following a well-tried approach, popular in the wearable computing research community, to provide a development platform for performing informal outdoor assessments of the software and the sensors. The author was aware during the development of the system that Remus would prove inadequate in meeting the requirements mentioned above, yet it was essential for it to be available for early testing in normal operating conditions.

4.5.2 Romulus final design

While development of the wearable AR application described in Chapter 5 was carried out on Remus and laboratory desktop systems, the mini-ITX form factor, introduced by VIA, became popular with PC enthusiasts. Its small size (17×17 cm), compared to standard ATX motherboards, limited power consumption (<30 W) and the fact that it is an almost fully-featured PC made it a popular basis for small, semi-portable devices such as routers and media-players. The mini-ITX specification was more promising than other SBCs presented in Chapter 2 and therefore was used in the Romulus wearable computer (fig. 4.2). PC/104 models did not have adequate 3D graphics hardware acceleration, mostly using the S3 Savage4 chipset[3] for which support was limited in Linux. EBX designs were expensive and required bulky and complex back-planes. At the time, SBCs of PCI card form factor based on Intel chipsets, also supported in Linux, were not available.

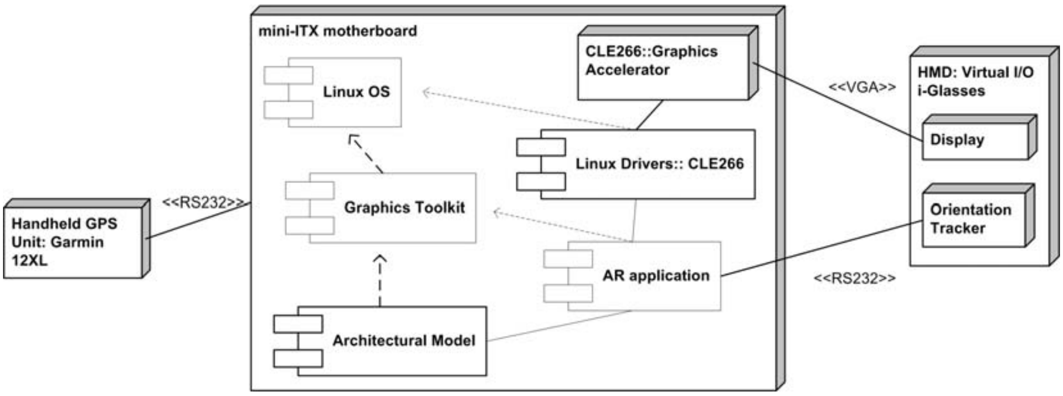


Figure 4.2: Final design UML deployment diagram

The EPIA series of mini-ITX SBCs from Via[4] includes various models with VGA/S-Video/Composite TV-Output, USB, a pair of RS-232 interfaces and wired Ethernet, integrated AGP graphics with 2D/3D hardware acceleration (CLE266 chipset), audio, and a single PCI slot for expansion.

[3] http://www.s3graphics.com/

[4] http://www.viavpsd.com/index.jsp

The series of motherboards satisfied almost all the aforementioned system requirements (Section 4.4), apart from size. Although mini-ITX boards are small compared to other desktop-oriented variants, they are still almost twice the area of other embedded boards. Nonetheless, their features, and particularly the hardware acceleration for 3D graphics, made these boards an appealing solution. The size compromise would be addressed with careful component arrangement in the wearable computer's vest.

In addition to the features the EPIA-series SBCs have, the author felt that the ever-increasing community of developers, enthusiasts and researchers that use mini-ITX boards would be a useful source of information for future developers[5,6]. The amount of reference material on mini-ITX surpasses many other small-form factors, providing support for problems encountered during development. Furthermore, the popularity of the platform gave rise to the expectation that it will evolve into more powerful configurations. As has been mentioned, off-the-shelf component evolution is governed by the marketplace [47], therefore when developers require prototype designs to be upgradable and future-proof it is advisable to choose popular platforms.

Motherboard

The highest-specification EPIA mini-ITX board has a 1 GHz C3 CPU. It uses a Via CLE266 chipset with Linux DRI (Direct Rendering Infrastructure) (see Chapter 5) drivers supporting hardware acceleration. Its two serial ports allow interconnection with the GPS and HMD tracker units. ATX power connectors and standard PC/AT layout allow the use of standard PC-compatible components. The motherboard's cost is fairly low ($< \pounds 150$). An estimate of the power consumption, based on informal assessments in the online mini-ITX developer community,[7] was expected to be 25–30 W with a hard disk and 512 MB of RAM.

Memory, storage and peripherals

512 MB of low profile DDR-SDRAM memory was considered adequate for this task. A spare 2.5-inch laptop hard disk of 20 GB was considered adequate storage medium (to be replaced by flash memory in future versions). A Netgear[8] 802.11g WG111 Wireless LAN USB dongle, based on a Prism chipset, was chosen for the wireless interface due to its driver support in Linux.

[5] http://www.mini-itx.com/

[6] http://www.linitx.org/forum/

[7] http://www.linitx.org/forum/

[8] http://www.netgear.co.uk/

Power regulation and batteries

A LEX[9] ATX DC-to-DC converter unit (max. 60 W) was employed for supplying the correct ATX signals on the motherboard from a 12 V source. The system would be powered from a pair of custom-made 12 V NiMH rechargeable batteries. These were preferred to commercial batteries due to their low cost ($\approx \pounds 35$ per pack). Ten 1.2 V cells were to be arranged in series, resulting in an overall capacity of 3700 mAh and allowing the pack to have the desired shape (i.e. flat) for easy integration with the vest. A custom charger was also designed (described in Section 4.6.4).

Head mounted display power supply

The Virtual i/O I-glasses HMD specification has been described in Chapter 2. The HMD requires a 9 V supply and a serial (RS-232) interface for the orientation tracker. A 9 V (2 A) regulator was designed to convert the 12 V battery supply (see Appendix C). The output of the regulator is connected directly to the HMD driver box.

Enclosure design

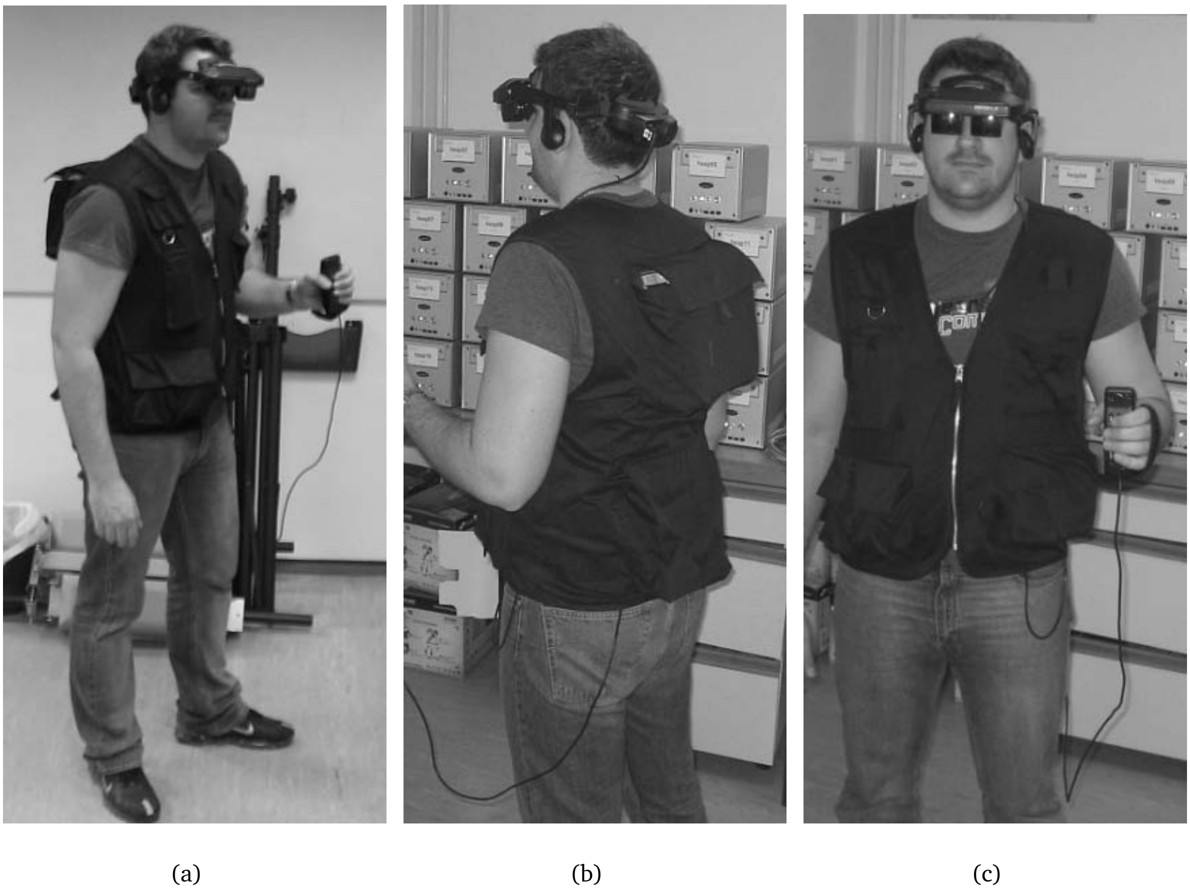
The case needed to encapsulate all the aforementioned components, cabling and external connectors while maintaining adequate airflow. The mini-ITX boards use standard aluminium I/O port side-plates, similar to desktop motherboards, simplifying considerably the enclosure design and allowing future motherboard upgrades. A square case was designed, to be made from folded aluminium, with a slot for the mini-ITX I/O panel on the side (schematic in Appendix E). To allow for adequate airflow and heat extraction from the case, ventilation holes were used. The hard disk had to be bolted on the top of the case, above the motherboard. Due to space constraints in the case, indicators for power, HDD activity, an on/off vandal-resistant switch and the second serial port connector were to be placed on the back plate of the wearable computer.

Vest design

The vest design was based on the notion of distributing the components of the system around the user's body. A prototype design was drawn (Appendix D), with the mini-ITX enclosure, being the largest and heaviest component, placed on the back of the user. All cables were to be routed in the interior of the vest. Each pocket required a cable-access hole on the back, with the main wearable pocket having two, to accommodate the number of connections to the mini-ITX board.

[9] http://www.mini-itx.com/

The finalised wearable is shown in figure 4.3, followed by a schematic overview of the Romulus architecture (Section 4.5.3) and a specification summary (Section 4.5.4).



(a) (b) (c)

Figure 4.3: The author wearing the Romulus wearable

4.5.3 The 'Romulus' final-design architecture

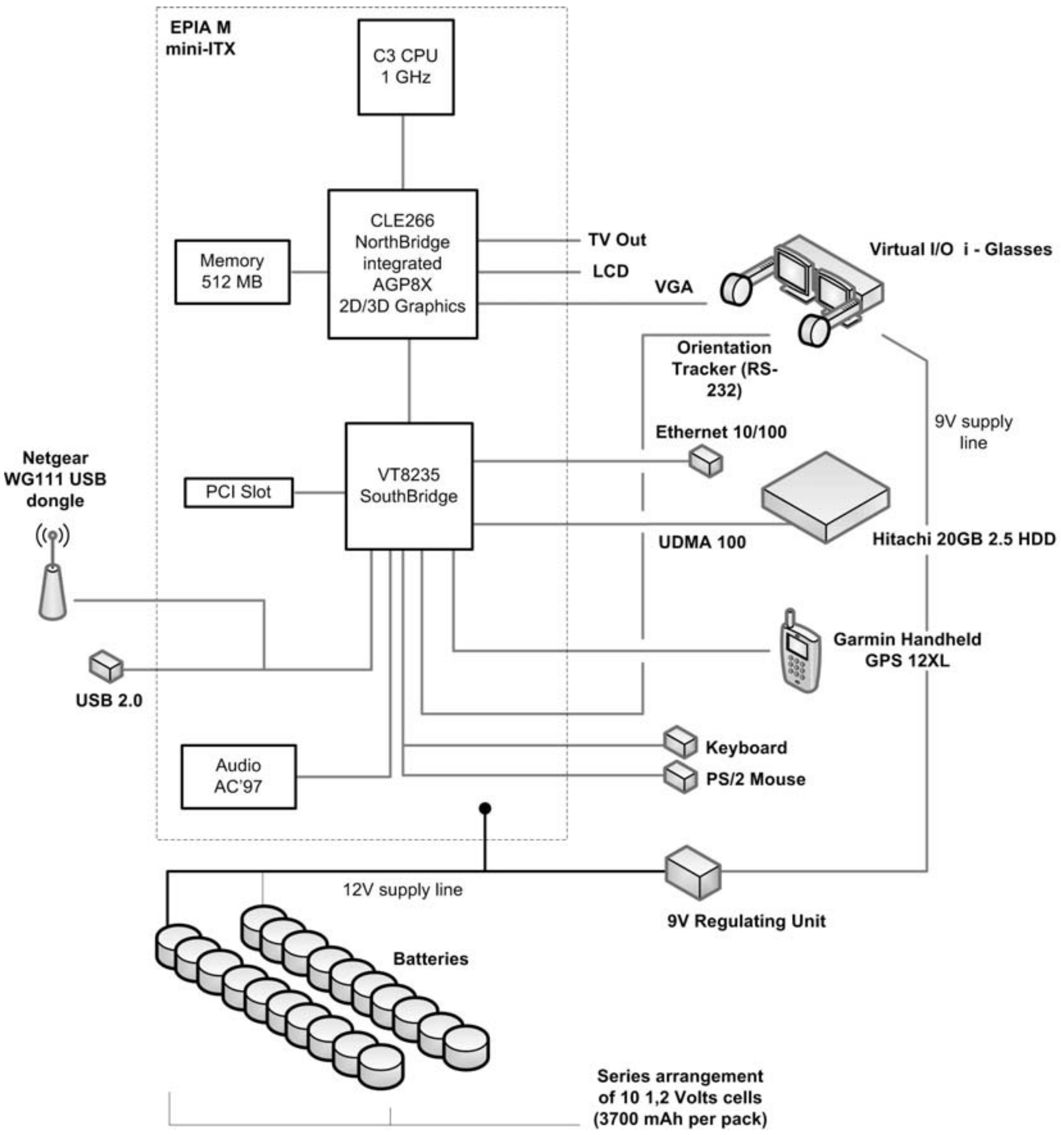


Figure 4.4: The Romulus architecture

4.5.4 The ‘Romulus’ wearable computer finalised specification

Main Components	
CPU board	Via Epia M 1000 Nehemiah Mainboard
PSU Board	Lex 12 V DC ATX Power Supply Board — compatible with Mini-ITX

I/O Devices	
Output	Virtual I/O i-Glasses
GPS	Garmin GPS12XL GPS Unit
Wireless LAN	NetGear WG111 USB adapter

Specification	
CPU	C3 1 GHz CPU
Memory	512 MB of DDR-RAM
Chipset	VIA CLE266 NorthBridge
Sound	VIA VT1616 6 channel AC’97 Codec
Ports	2 Serial Ports, 1 Parallel, 2 USB, PS/2 (K/M), IrDA, IEEE 1394 ‘Firewire’
Network	Ethernet 10/100 — Wireless LAN (802.11) via Netgear’s USB adapter
Expansion	PCI interface
Storage	20 GB HDD - Hitachi
Power	12 V DC input — secondary output 9 V@1W output (HMD supply line)
Batteries	2× NiMH custom made battery packs (12 V)
Operating System	RedHat 8.0 with DRI accelerated driver, v.40038
Weight	4.325 kg (including vest, HMD, batteries and handheld GPS)

4.6 Wearable computer construction

4.6.1 Mini-ITX motherboard

The construction of the Romulus wearable was fairly straightforward as it did not require any custom electronic circuitry apart from the 9 V regulator. Since the board is a miniaturised version of a standard ATX motherboard (fig. 4.5), most of the associated cabling and connectors are similar to the ones used in desktop PCs. All the cabling required to power the system is provided with most mini-ITX power supply units.

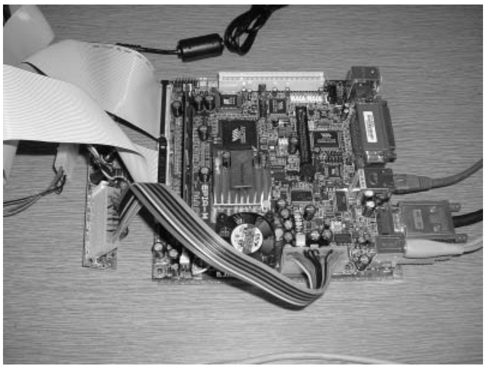


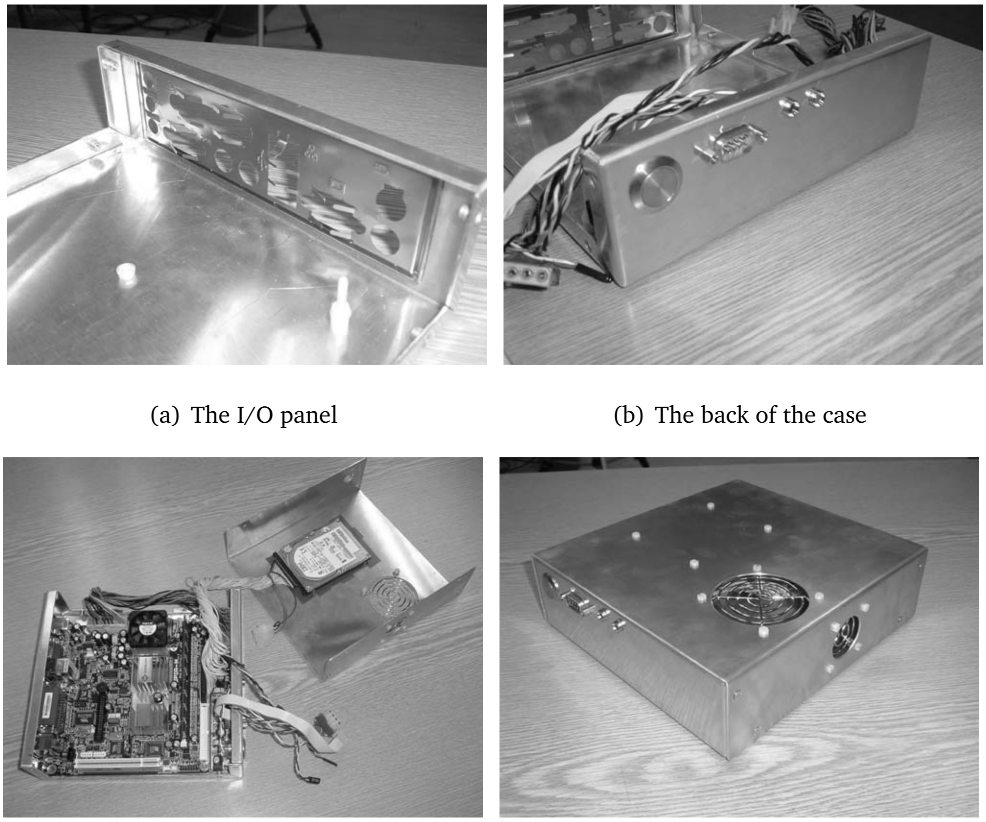
Figure 4.5: The mini-ITX motherboard

A Hitachi 2.5-inch HDD was used as a system disk. The motherboard has a standard 3.5-inch IDE connector, therefore a 2.5-to-3.5-inch IDE adapter was required. A vandal-resistant mechanical switch was used for the power button and a pair of LEDs for the power and hard disk activity indicators. Connecting them to the board was done with a set of cables suitable for a standard ATX desktop case, as the required connectors are the same. All the interfaces of the board, apart from the second serial port which requires a separate connector, are on its front side, eliminating any need for extensive drilling.

The second serial port connector resides on the motherboard and requires a custom adaptor to reach the back of the wearable. Quite strangely, Via did not include a standard serial port pin-out on the board, allowing universal connectors to be used. Instead all pins are numbered from 1 to 9 following the numbering of a DB9 serial connector, i.e. pin 1 on the board is pin 1 on the DB9 connector, pin 2 is DB9 pin 2 etc. (Appendix C).

4.6.2 Case

The case of the wearable is built from folded aluminium (fig. 4.6) (Appendix E). The front face of the case has a slot where the supplied I/O panel is placed, making upgrading the motherboard with other mini-ITX boards as straightforward as on desktop PCs (fig. 4.6.a). A custom-made adapter for the second serial port was constructed and bolted on the back of the case, near the LED indicators (fig. 4.6.b). The case has a pair of circular holes, one with a diameter of 60 mm on the top side of the lid of the case and the other, with a diameter of 40 mm, in the side near the CPU (fig. 4.6.d). A 40 mm fan was placed on the side hole to extract heat, allowing sufficient airflow to keep the CPU relatively cool. A set of grilles for both ventilation holes completed the enclosure.



(a) The I/O panel

(b) The back of the case

(c) The Romulus wearable components

(d) The case closed

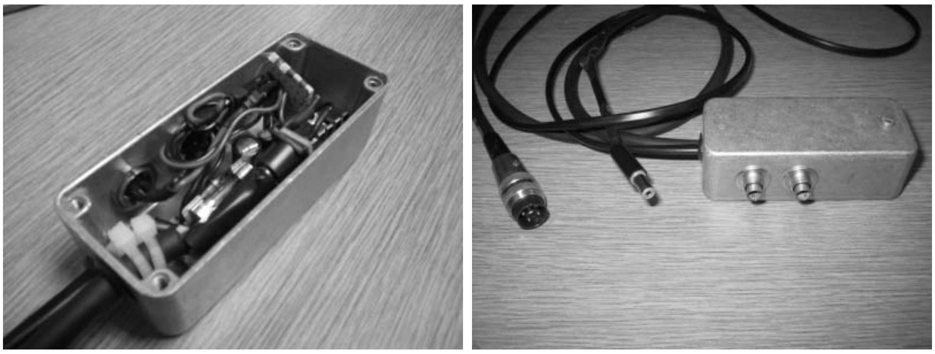
Figure 4.6: Construction of Romulus

4.6.3 Additional circuitry

A voltage regulator supplies the Virtual I/O driver box with its 9 V power rail. A separate, small case holds all the circuitry and serves as a heat-sink for the regulator, which was bolted to the case in order to dissipate heat more efficiently (fig. 4.7.a). One of the outputs is connected to the Virtual i/O HMD driver box and the other, effectively supplying 12 V directly from the batteries, was connected to the wearable via a lockable connector (fig. 4.7.b). A pair of lockable connectors of smaller diameter was also used to connect the batteries on the side of the regulator's case. The construction of a separate box was necessary due to the limited space in the main case. Furthermore, a separate power supply case simplifies an upgrade to a more advanced regulating system, perhaps one that will provide battery monitoring as well.

4.6.4 Battery pack

The battery pack is assembled from twenty 1.2 V NiMH rechargeable cells with a capacity of 3700 mAh each, which is adequate for an operating time of an hour per pack, as the current



(a) The 9 Volt DC-to-DC box with its internals visible (b) The 9 Volt DC-to-DC box and connectors

Figure 4.7: The DC-to-DC regulator

drain of the system was measured at about 3 A. The cells were arranged in series, forming two separate packs. These were subsequently connected in parallel (fig. 4.8) to form a dual-pack battery assembly, in order to increase overall capacity. Each pack is connected via the lockable connectors on the small power regulating unit.

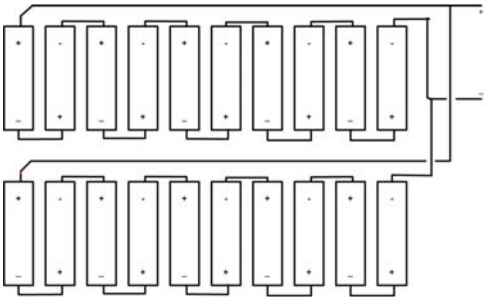


Figure 4.8: Battery circuit schematic

The chosen cells were manufactured by GP.[10] A thermistor of 73°C and a resetable fuse of 9 A were also connected in series between the middle cells in order to protect the batteries from short circuit during assembly. The packs were wrapped with insulating tape (fig. 4.9). Care was taken so that, when each cell was soldered with the next, the soldering iron does not damage the plastic protective shell thus exposing the metallic casing underneath. The positive tag must not come into contact with the metallic casing (which is connected on the negative pole) as this may damage the cell.

[10] `http://www.gpbatteries.com.hk/`

Figure 4.9: The two battery packs assembled and wrapped in insulating tape

Battery charger

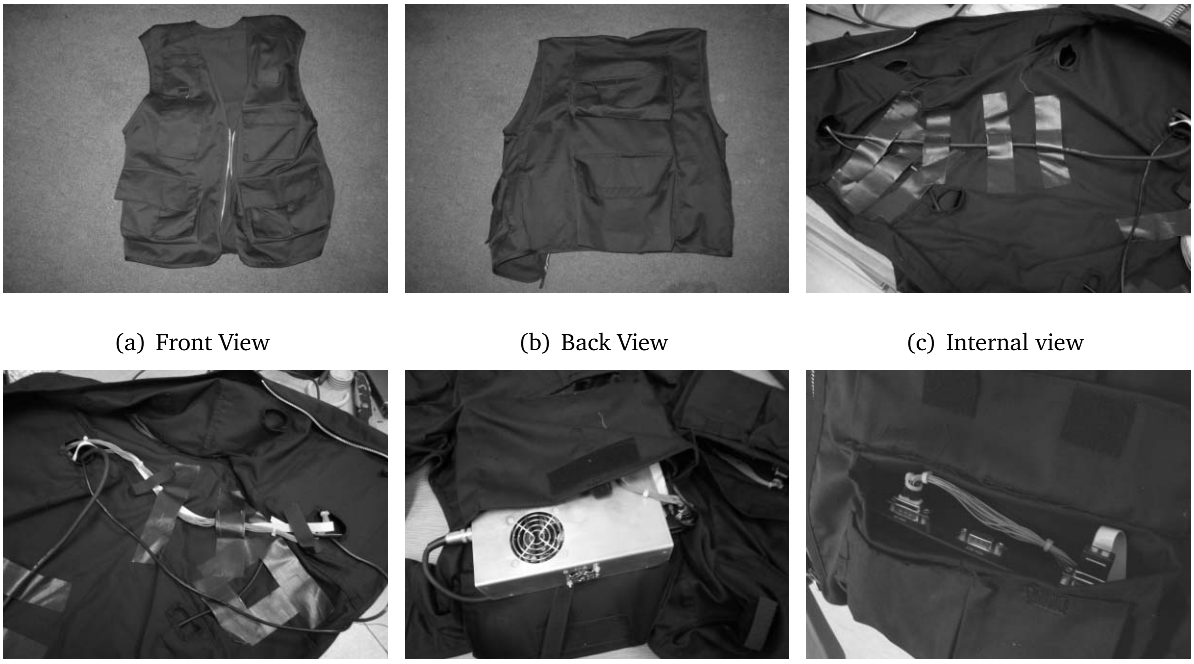
The battery charger is a simple circuit based on a LM317 voltage regulator (see Appendix C). The resistor in the circuit required calculation in reference to the charging current of each battery pack. Since each battery has a charging current at 370 mA (1/10 of its capacity in mAh) and given that Vref is 1.25 V (from data-sheet of manufacturer) the resistor needs to be **3.3** Ω. Care needed to be taken so that the resistor was able to accommodate the high current value. The resistor power dissipation is $P = I^2R$, so the resistor must be of at least 0.5 W rating.

4.6.5 Vest integration

A photographers vest (fig. 4.10) was modified[11] by adding a large pocket at the upper half of the back to carry the mini-ITX enclosure and various holes on the back of the pockets to route the wiring (see Appendix D). A secondary pocket of similar size was also placed lower on the back for future expansion and a storage area. The batteries and HMD driver box were placed on the front pockets of the vest.

The GPS unit was placed in the front chest pockets, with the GPS antenna mounted on the epaulet. Due to the weight of the case the pocket is reinforced with velcro tapes, to ensure the enclosure stays firmly in the proper place. All the wires were routed through cloth tape loops to hold them in place (fig. 4.10.c/d). Overall weight with components mounted on the vest and including the HMD, batteries and handheld GPS is 4.325 kg.

[11] Modifications were made by Isabella Ritsou.



(a) Front View

(b) Back View

(c) Internal view

(d) Internal view

(e) The enclosure in the back-pocket

(f) The Virtual i/O driver box

Figure 4.10: The Remus wearable computer vest

4.7 Operating system installation and driver configuration

Windows and Linux were installed in a dual boot configuration; the choice being RedHat Linux 8.0 and Microsoft Windows 2000. The reason the system was initially built with a dual-boot configuration was to explore the performance of 3D graphics under both operating systems. Windows was subsequently dropped for reasons described in Chapter 2 and because it did not offer any performance advantages. Drivers for graphics, sound, USB and the Ethernet adapter for Linux were downloaded from the ViaArena website[12] and installed.

Of great importance to the performance of the system was the introduction of Linux drivers supporting DRI[13] (see Chapter 5) during the summer of 2003. The XF86Config configuration file was modified to activate DRI as a module during power-up. The GLUT packages for OpenGL code development (see Chapter 5), required in this application, were downloaded and installed. Likewise, a source package of the driver for the Netgear WG111 USB adapter was downloaded[14] and installed. No other configuration settings were required for any of the drivers. Finally, the permissions on the serial ports were changed to be accessible from user processes.

[12] http://www.viaarena.com/

[13] http://dri.sourceforge.net/

[14] http://www.linux-wlan.com/

4.8 Conclusion

The main purpose of designing and constructing Romulus was to result in a hardware platform that could be practically used in *in-situ* experiments for wearable AR while being comfortable to wear, easy to duplicate and relatively cheap to construct. It is based on a modern embedded board, the mini-ITX. The choice of this particular type of board was made principally due to its hardware-accelerated graphics controller and its low-power configuration.

Although the design and implementation of Romulus is fairly simple and does not require any particular hardware expertise, apart from the one required to assemble desktop PCs, it aims to maintain an adequate level of performance in rendering a relatively complex 3D scene. Romulus was used as the development platform for the Gosbecks wearable AR tour guide application presented in Chapter 5, as well as for field testing of the system's accuracy and usability discussed in Chapter 6. As will be presented, the performance rendering requirements were met, relying both on Romulus's hardware-accelerated graphics and software speed improvements.

Chapter 5

The Augmented Reality Tour Guide Application

5.1 Introduction

This chapter describes the design and implementation of the Gosbecks archæological site tour guide application, built to be functional on the Romulus wearable and with main purpose of realising an *in-situ*, outdoor experiment for wearable AR. Section 5.2 presents an overview of the application, in conjunction with the overall system, pinpointing some initial design considerations, practices and constraints. Section 5.3 describes the needs requirement and a conceptual design of the application. The mandatory components of this research effort are also presented. A detailed requirements specification is provided in Section 5.4. The finalised design is presented in Section 4.5. The section describes the architecture of OpenGL, its callback mechanism and the impact this had on the application's final design. Particular attention is given to the view mechanisms of OpenGL and their interface with the location and orientation sensors. The finalised application structure is also described. Section 5.6 presents the Gosbecks model, providing an architectural description and leading to the implementation of the architectural elements in OpenGL. Section 5.7 presents the first version of the Gosbecks tour guide, implemented in 2001. It describes the GPS and HMD tracker software interfaces and the integration of these components into the overall application. It also describes the initial tests held in the beginning of 2002 with Remus and presents the problems identified in these tests and their proposed solutions. Section 5.8 presents the final version of the application, incorporating speed optimisation techniques such as scene-graph organisation, view frustum culling (VFC) and levels of detail.

Section 5.9 shows the improvement in rendering speed achieved with these enhancements. Section 5.11 summarises the tour guide application and sets the scene for the experimental work described in Chapter 6.

5.2 Overview of the application

The aim of the application described in this chapter is to provide a wearable augmented reality tour of a Roman temple *in situ*, employing a 3D reconstruction of the temple. The application is to run on the Romulus wearable computer, equipped with the global positioning system and a HMD head tracker for location and orientation determination respectively.

As discussed in Chapter 3, the main challenge of realising the above is that the application needs to accommodate the limited rendering capabilities of wearable systems. As pinpointed by Baillot *et al.* [24], Gleue *et al.* [80] and Piekarsky [165, 169], the main challenge is overcoming this rendering bottleneck of wearable computers in achieving realistic 3D environments, while maintaining adequate frame rates. The problem is more severe in wearable AR than normal AR applications, where rendering is not so much an issue [19].

However, apart from devising powerful wearable computing hardware, proper utilisation of 3D graphics toolkits is required in order to exploit the rendering capabilities of any particular platform. As Piekarsky describes [165], software for outdoor wearable AR must be efficiently designed and be able to run on hardware that may be a number of generations behind the current state of the art of desktop computers. Correa and Marsic [57] further add that interactive applications on mobile devices require a series of speed improvements, such as scenegraphs (see Section 5.8), in order to be functional under certain performance thresholds. They characterise the relationship between software enhancements and hardware capabilities as 'combinatorial optimisation'.

The system described in this chapter is intended to fulfil two specific aims. Firstly, it provides the software component which forms the basis for the evaluation work described in Chapter 6. Secondly, it provides a wearable AR framework being independent of the model being presented to the user and, as far as can be achieved, independent of the underlying hardware, the latter by virtue of a platform-independent rendering toolkit. Unlike the Touring machine [71], which does not employ demanding 3D reconstructions, and the Tinmith project [167, 169, 170, 171], which focuses on dynamic content creation and the user interface, the system described here aims to provide an architectural wearable AR tour that focuses on rendering a complex scene quickly (see Section 5.4). The ultimate purpose is to built an application framework that will allow further additions, such as orientation and localisation input filtering, additional 3D scenes and a 2D information interface while maintaining an adequate level of performance (see Section 5.4). The Archeoguide project [235] has similar aims but it fails to provide true 3D reconstructions, instead using pre-rendered images of the reconstruction from specific viewpoints.

5.2.1 System architecture

The system described follows a distributed approach, unlike the centralised one used in Archeoguide [235] (see also Chapter 3). The reasons for this choice are:

- The site in question is under the protection of English Heritage, which does not allow the ground to be disturbed, such as by erecting further signage, radio masts or the server, required for a centralised architecture. In addition, independence from environmental constraints allows the system to be used with other 3D reconstructions, without requiring a major architectural overhaul.

- A distributed approach supports a large number of users because the data communication requirements remain largely independent of the available bandwidth; there is no central server. All the processing is done on the mobile units.

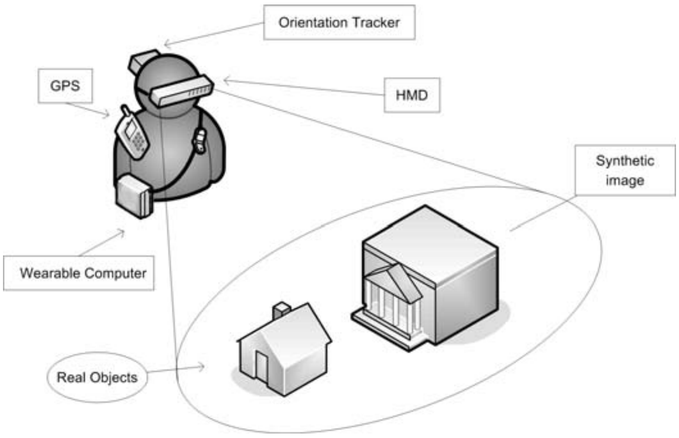


Figure 5.1: Concept drawing of system functionality

Each node of the system's architecture is a wearable computer, equipped with a handheld Garmin 12XL GPS unit connected to a serial port and the Virtual I/O HMD with integrated orientation tracking, likewise connected to a serial port (fig. 5.1). All rendering of the model is performed on the wearable and not on a centralised server. This supports an arbitrary number of users as there are no bandwidth limitations as in the case of a centralised system. On the other hand, rendering 3D information of the complexity of the Gosbecks model (see Section 5.6) requires more processing power than usually found in wearable computers. The Romulus prototype addresses to some extent the hardware issues, as discussed in Chapter 4.

5.3 Requirements and conceptual design

This section presents a summary of the application requirements, analysed in detail in Section 5.4, picking up from those related to hardware, presented in the previous chapter. The requirements are:

Requirement 9: The application to be implemented has to run on the Romulus wearable computer and has to provide a wearable AR tour of a 3D architectural reconstruction of the Gosbecks main temple.

Requirement 10: The application has to provide a frame rate of at least 10 fps, for a model of about 30.000 polygons (model specification provided in Section 5.6), while rendered on the Romulus wearable.

Requirement 11: Orientation and location information has to be extracted from the Virtual I/O HMD and the Garmin handheld GPS respectively (presented in detail in the next section) via the two serial ports of Romulus and control of the toolkit's viewpoint. Therefore, the toolkit for creating this tour guide application has to accept alternative, to a keyboard or a joystick, input mechanisms and to provide adequate control for manipulating the view of the virtual world.

Requirement 12: The application has to provide a positional accuracy of 2 metres and an orientation accuracy of less than $0.5°$. These values are derived from previous research with these modalities [111, 153, 157].

Requirement 13: Due to the use of the Virtual I/O HMD for visual output, the application has to render scenes at 640×480 pixels, at 60 Hz.

Requirement 14: The application it must flexible as far as content creation is concerned, so that the model primitives look fairly realistic and are easily distinguishable and identifiable.

Requirement 15: The application needs to remain independent of the 3D model, to allow reuse with additional scenes.

5.3.1 Mandatory system components

The next sub-sections present the mandatory components, used in this research. These are the Garmin 12XL handheld unit and the Virtual I/O I-glasses. Their use was directed from the fact that they were readily available in the author's laboratory, thus reducing the overall cost and allowing replication.

Global Positioning System

The Global Positioning System (GPS) is a system able to determine position on the Earth. Twenty four satellites orbit above the Earth and are continuously monitored by five ground stations located worldwide (fig. 5.2). Using a GPS receiver, satellite signals can be detected and location can be determined with relatively high precision by the method of 'trilateration'[1]. Position can be calculated from distance measurements (ranges) to at least three satellites, whereas at least a fourth measurement is used in practice to compensate for clock discrepancies and other inaccuracies. These measurements can be processed to give the receiver's position on the Earth, in terms of latitude, longitude and altitude, with relatively high precision for most navigation tasks.

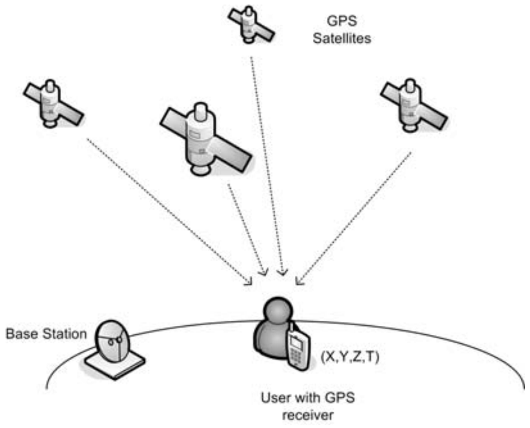


Figure 5.2: The Global Positioning System

GPS receivers can be found in ships, cars and aircraft for navigation purposes. The system

[1] Contrary to triangulation, where the calculation involves angles and distances, trilateration involves only distances [204].

was originally developed by the U.S. Department of Defence and is used extensively by the military under the name NAVSTAR. Modern commercial receivers come in various sizes and capabilities. Most popular are handheld models that are slightly larger than a mobile phone and provide basic GPS functionality, such as the Garmin handheld unit used in this research effort (fig. 5.3).

Figure 5.3: The Garmin GPS12XL handheld unit used with Romulus

Nonetheless, although a modern basic GPS system can generally give a positional accuracy of about 10-20 m[2] [99], more people seek higher precision. An improved version of the basic system is called Differential GPS (DGPS); this can improve the accuracy of position determination to 1-5 m when moving and even less when stationary. The principle of DGPS is based on the use of a second GPS receiver, at a known position, being used as a reference point to correct the mobile unit's errors. These errors exist due to ionosphere and troposphere delays, signal multipath due to buildings or terrain, receiver clock errors, orbital errors (ephemeris errors), and the number of satellites visible as well as their relative positioning (Satellite geometry/shading)[3].

Assuming that the mobile and the stationary (reference) are within a few kilometres of each other, the signal travelling from visible satellites travels through the same atmospheric slice, therefore introducing virtually the same errors. These can be multi-path errors, occurring around the receiver and receiver errors, unique to each unit. The overall error in position can be computed to the base station and subsequently transmitted to the mobile unit to correct its position (fig. 5.4). Transmission of DGPS correction information is also done by various agencies nowadays and requires a radio receiver on the end of the mobile unit to intercept them (usually at 300 kHz). Many units found in the market have such receivers, enabling DGPS functionality, yet their cost is high.

[2] source: http://www.garmin.com/products/gps12xl/spec.html

[3] http://www.garmin.com/aboutGPS/

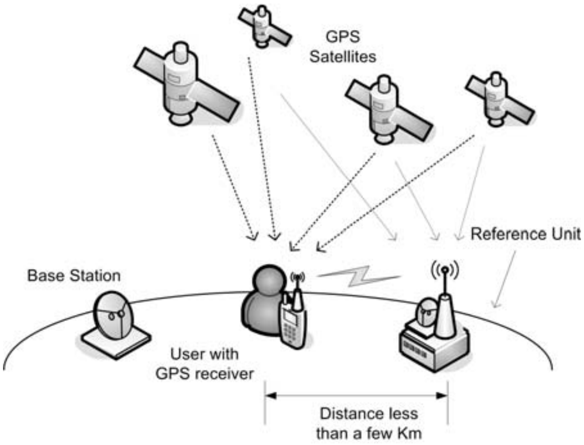


Figure 5.4: Differential GPS

However, many people claim various different magnitudes of error, depending on the application, system configuration and even geographical position. Schwieger [198] claims that, using handheld GPS receivers an accuracy of 10 cm is achievable under favourable conditions, whereas 1-5 m is more common. Ardo and Pilesjo [12] report far greater errors, in the order of 20 m and more, using a Magellan[4] GPS NAV 1000.

On the wearable computing field, Feiner *et al.* [71] report that the Touring Machine wearable, using a subscription-based DGPS system had an accuracy of about 1 m. Thomas *et al.* [223] claim that their Phoenix 2 wearable computer, equipped with a Trimble SVeeSix-CM3 GPS and a Aztec RDS3000 Differential GPS Receiver Module achieved an accuracy of 5-20 m. Ashbrook and Starner [13], while experimenting with a Garmin 35 LVS receiver obtained an accuracy of about 15 m. Early incarnations of the Tinmith hardware, using a Garmin 12XL GPS receiver had an accuracy of 2-5 m, as claimed by Piekarsky *et al*. [172]. In other cases, the same authors claim that the accuracy of their system was 5-10 m, using commercial units, improving up to 50 cm when using DGPS [173]. Piekarsky and Thomas [168] also add that their GPS implementation, based on an Intersense IS-300 GPS unit, exhibits unpredictable behaviour, making it the most critical component in their registration mechanism. They also add that even with this level of accuracy the system is far from ideal for wearable AR [168]. The system described here, based on a Garmin Handheld 12XL unit, has a claimed accuracy of about 2 m [111] when more than 4 satellites are 'visible', as will be demonstrated in the experiments, described in Chapter 6.

[4] `http://www.magellangps.com/en/`

The Virtual I/O HMD orientation tracker

Orientation determination is provided by the Virtual I/O I-glasses HMD (see Chapter 2). The existence of a 3DOF (degree of freedom) head tracker is also an important reason for using this HMD and, although its output in terms of quality is not as good as some modern HMDs, it is still adequate and the author considers it still one of the best solutions for AR prototypes. The I-glasses have been used in the past in wearable computing by Billinghurst *et al.* [35] in their Wearable Conference Communication Space, by Fickas *et al.* [72] and Kortuem *et al.* [126] and the NETMAN wearable computer and by Reitmayr and Schmalstieg [186]. Furthermore, Thomas *et al.* [222] have used it in various flavours of the Tinmith wearable and Feiner *et al.* used it in the Touring Machine [71].

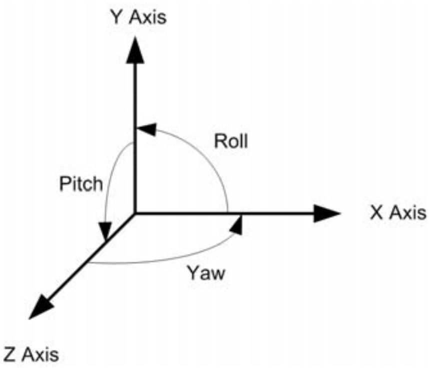


Figure 5.5: The I-glasses head-tracker coordinate system

The I-glasses head-tracker sends yaw, pitch, and roll information to the host computer via a serial link (RS232) at 19200 bps. Positive yaw is defined as a left head rotation, positive pitch as an upward head tilt and positive roll as a left head tilt (fig. 5.5).

5.3.2 Hardware drivers,Mesa3D and Direct Rendering Interface

OpenGL along with Glut have the potential of satisfying the aforementioned application requirements, along with platform independence (for future versions of the application), wide support in Linux and a vast online community of developers. OpenGL, widely used in modern 3D games is extremely flexible when it comes to content creation and is widely supported from hardware graphics accelerators, exploiting the potential of the latter through techniques such as display lists, vertex arrays etc. In addition, manipulating the viewing mechanism can be done in a number of ways (see Section 4.5), allowing the use of the GPS and HMD modalities as viewpoint control mechanisms.

One of the most important advantages of OpenGL is its support for hardware acceleration. OpenGL is supported by almost all modern graphics cards and chipsets, yet the benefits of hardware acceleration and performance benefits are down to the graphics drivers. As far as Linux support is concerned some vendors issue their own, proprietary drivers whereas others rely on the Direct Rendering Infrastructure (DRI). This is a framework that allows direct access to graphics hardware, its primary purpose being to create fast OpenGL implementations for Linux-based systems. DRI has been an integral part of XFree86 from version 4.0.0 onwards, with a requirement of a kernel later than 2.4.0 and it integrates with Mesa[5], an open-source implementation of the OpenGL API. A number of chipsets are supported by 3D-accelerated drivers, including those produced by ATI, Matrox and 3dfx. During August of 2003, support for the Via CLE266 chipset was also introduced, enabling the system described here to use hardware acceleration for 3D rendering.

The DRI framework allows programs to communicate directly to the graphics card. Applications can send commands directly to the graphics card, performing fast, hardware-accelerated rendering while also freeing the CPU from the associated processing overhead. This process is known as *direct rendering*. Without the DRI framework, applications have to perform the rendering purely in software. This imposes a significant overhead on the CPU, resulting in low frame rates.

DRI has paramount importance in the rendering speed of the system described here. Initial tests were done without any DRI support, relying only on the software rendering of Mesa3D. Once DRI drivers were available the performance boost was significant, as will be discussed in Section 5.8.

5.3.3 Application architecture

The preliminary design of the application consisted of two main components:

- the 3D architectural model of the temple;
- a framework to display 3D models, using real-time viewpoint manipulation from location and orientation sensors.

The application was separated in four discrete sub-modules. The core application was to be a simple program encompassing functions for rendering a 3D model, but not the model itself.

[5] `http://www.mesa3d.org/`

Location and orientation information is processed from separate subsystems. The fourth module contains the model. A conceptual deployment model using UML notation [31, 32], depicting the main elements of the preliminary design is show below (fig. 5.6). The next section provides a detailed specification of the software requirements [106].

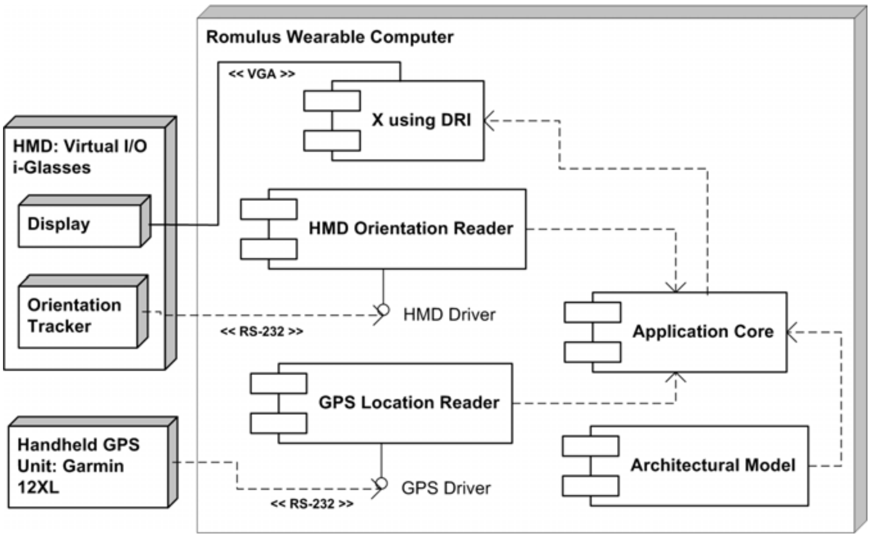


Figure 5.6: UML Deployment diagram of the Gosbecks tour guide application

5.4 Software Requirements Analysis

5.4.1 External interface requirements

User interfaces

The application needs to provide a visual output of 640×480 at 60 Hz for the HMD. No input interface is required since the application reacts only to changes in the user's orientation and location (supplementary information such as positioning, number of objects rendered, number of facets per cylinder and frames per second can be overlaid on the HMD's view). After the system is powered on the application should start automatically in full-screen mode. The field of view used for rendering needs to match that of the HMD so that objects at various distances appear with the correct proportions [148].

Hardware interfaces

The application was to be interfaced with the Virtual I/O head tracker and the Garmin GPS 12XL through a serial ports at the maximum possible connection speeds allowed from their respective drivers, i.e. 19200 bps for the HMD and 9600 bps for the GPS. The information extracted from these interfaces (yaw, pitch and roll in degrees from the head tracker and latitude and longitude from the GPS) need to be translated to a more usable format; more precisely, values from the HMD needed to be translated to radians, for use in trigonometric functions, while data from the GPS needed to be converted to local X and Y. Altitude was omitted for simplicity and because the Gosbecks field is relatively flat[6]. Visual output is done through the VGA interface at the aforementioned resolution. The graphics driver needed to support DRI (see Section 5.3) for hardware-accelerated rendering, and the application in turn needed to exploit that acceleration to provide the desired frame rate.

Software interfaces

The application, is written to use the OpenGL and Glut APIs. Mesa serves as the OpenGL core for the open-source XFree86/DRI OpenGL drivers found in RedHat 8.0, which was the operating system of choice. Glut is an integral part of Mesa in the RedHat Linux distributions.

The GPS pipeline starts with `gd2.c`, a C procedure that communicates with the Garmin GPS12XL GPS receiver via a proprietary protocol over the serial port. The code is available from:

`http://vancouver-webpages.com/pub/peter/gd2.tgz`

The head-tracker pipeline starts with a modified[7] version of the C code available at:

`http://iglasses.weirdoz.org/`

The new driver for the serial port was integrated with the software. The module initialises the tracker and reports the orientation of the HMD in the form of yaw, pitch and roll, expressed in degrees, requiring as mentioned, subsequent processing for use in the orientation subsystem.

[6] However there was provision for its addition in the future.

[7] Driver partially modified by David Johnston and Eddie Moxey; remaining components modified by the author.

5.4.2 Functional requirements

In normal operation mode the application should react only to the user's movement and orientation changes. When the system is booted the temple model appears facing the user. Having a flexible temple placement in terms of local position allows the application to be evaluated on the university campus and not only at Gosbecks. A new viewpoint is calculated for each reading of the GPS and head tracker and the new view is projected through the HMD. Overall system delay should be less than 100 ms (i.e. frame rate above 10 fps). The user should be free to roam to any position in the temple. Because there are no remains on the actual field the user can 'pass through' 3D elements and collision detection is not implemented.

For development and debugging purposes, the system is required to be able to accept keyboard input to allow virtual object manipulation, for examination of misalignments and positioning errors, a switch between different values of facets per cylinder (6, 8, 16, 32 per cylinder), a switch between smooth and flat shading, wireframe and solid rendering and a reset and quit function. Options for using the application in 'desktop mode', with a choice of input methods, such as keyboard and HMD or keyboard only, are also required in order to use the application for demonstrations.

5.4.3 Performance requirements

The main performance requirement of the application is that it should render a model in excess of 30.000 polygons with a frame rate over 10 fps, with the location and orientation sensors in use. This frame rate includes the time required to read and process information from the GPS and HMD, coordinate conversion, viewpoint calculation and rendering the scene. Hence, this frame rate includes both sensor delays and the processing delay. Modifying the equation of Atkins [14], we have:

$$\text{Frame Period} = \text{HMD Delay} + \text{GPS Delay} + \text{Processing Delay} + \text{Rendering} \tag{5.1}$$

5.4.4 Model specification requirements

Although the model is not an integral part of the application, it is its visual focal point. Its specification affects directly some performance factors of the application. The user should be able to look at the virtual world and be able to distinguish elements of the temple easily. Therefore the amount of detail of the model needed to be such that the user is able to distinguish features such as the Ionic and Doric columns, capitals and bases, walls, doorways, roof and pavements. Colour was to be used to distinguish between features of different material, like ground, roofs and marble. However, producing realistic colours is beyond the scope of this work, therefore colouring specification is not exhaustive. Global scene lighting was to be kept as simple as possible and smooth shading was to be used to increase realism of cylinders, while maintaining the number of polygons as low as possible. During the design period it was estimated that the model would comprise of 25,000 – 30,000 polygons.

5.4.5 Design constraints

Although placement of the model in terms of latitude and longitude is flexible, orientation needs to be precise, with the temple entrance facing roughly East (14° anticlockwise turn from the North–South axis), because of the nature of the HMD tracker which always returns a value relative to north. The range of yaw is $\pm 180^\circ$, pitch $\pm 60^\circ$ and roll $\pm 60^\circ$.

5.5 Proposed design

The analysis requirements presented in the previous section led to the final design of the application, implemented around the viewing mechanisms of OpenGL,GLU and Glut, and the callback-driven event mechanism of the latter. This section provides some background on the way that OpenGL works and introduces notation and concepts that are used in discussions of the application.

5.5.1 The camera paradigm

The OpenGL viewing pipeline is easily explained using a camera analogy. The scene elements are a model and a camera, both able to be placed freely in 3D space. The first step is to apply any modelling transformations that place the elements of the model in the right positions to form the scene. The next step orients the camera so that it can 'see' the model. The camera then projects the 3D scene to a 2D image on the display. Moving an object uses so-called *modelling transformations*, while moving the camera is called the *viewing transformation* [203]. Their combination is usually referred to as the *modelview transformation*. OpenGL concatenates modelling and viewing transformations into a single matrix, while a separate matrix describes the projection transformation. The projection transformation defines the view frustum (a frustum is a truncated pyramid). The viewport transformation finally places the complete scene on the display (fig. 5.7).

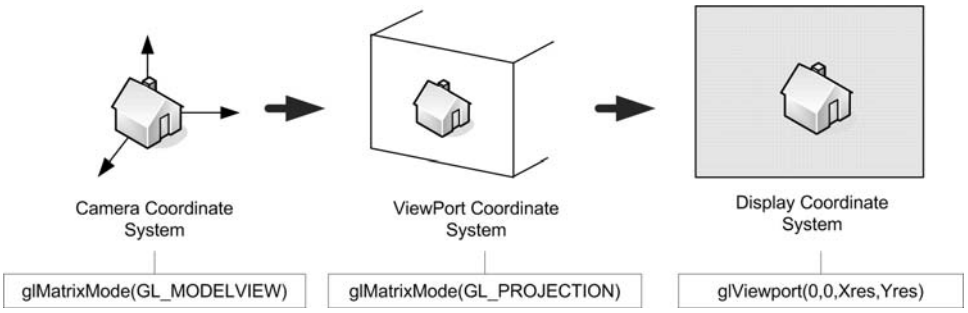

Camera Coordinate System
glMatrixMode(GL_MODELVIEW)
ViewPort Coordinate System
glMatrixMode(GL_PROJECTION)
Display Coordinate System
glViewport(0,0,Xres,Yres)

Figure 5.7: The OpenGL viewing pipeline

The camera can be specified by its position, the point it is looking at and a vector that describes its the orientation, these data being used as arguments to a GLU function called `gluLookAt()`:

$$\texttt{void gluLookAt(}C_x\ldots, V_x\ldots, U_x\ldots\texttt{)}$$

where (C_x, C_y, C_z) are the camera's object space coordinates, (V_x, V_y, V_z) define the point at which the camera is looking and (U_x, U_y, U_z) define which direction is 'up' (fig. 5.8). `gluLookAt()` computes the inverse camera transform according to its arguments and multiplies that onto the modelview matrix. The advantage of using `gluLookAt()` is that its arguments are easily obtained by the GPS receiver and head tracker.

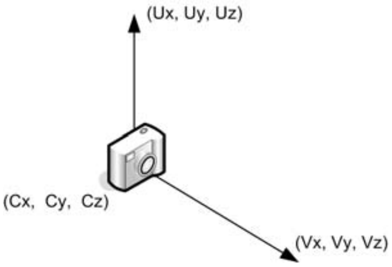


Figure 5.8: The OpenGL camera using `gluLookAt()`

5.5.2 The GLUT callback loop

GLUT is designed around the idea of an event-driven application. Programs using GLUT employ callback functions to structure event handling. These callbacks handle all the events to which the program responds, such as keystrokes and mouse-clicks. When an event occurs the system generates an associated callback, which in turn calls specific routines written by the programmer. If a specific callback is not registered for a particular event, no action is taken. Appendix G shows a flow-diagram of a typical GLUT program, with detailed information for each callback. Each callback responds to a particular event e.g. the reshape callback is performed in response to window resizes whereas the keyboard callback is for keystrokes from the user.

There are particular steps which are essential to a GLUT-based program. The first requires the initialisation of the GLUT library (`glutInit`). The second step requires setting up a window, with size, position and title as arguments, where the scene will be displayed. The third step is the ‘display’ callback — registered by `glutDisplayFunc` — where the actual drawing of the scene takes place. The program must register a display callback routine. Finally, GLUT’s event handling loop is entered by calling `glutMainLoop` (see Appendix G).

Although the aforementioned callbacks are essential, most programs use the ‘reshape’ and ‘keyboard’ callbacks as well. These are set up to respond to window resizes and user input from the keyboard or a mouse respectively. More important for this application is the capability of registering an ‘idle’ callback, which is invoked when the program remains idle, using `glutIdleFunc`. This callback is often used for animations. For that, the associated routine must invoke `glutPostRedisplay()` so that the window is refreshed by calling the display routine in the next loop pass. In this application the ‘idle’ callback is used to read the GPS and HMD data and cause the scene to be redrawn, taking into account changes in the user’s position and viewpoint in `gluLookAt()`.

5.5.3 Application structure

The following diagrams depict the application functionality and component design. A detailed UML [32] deployment diagram presents the sub-components of the application and their interdependencies (fig. 5.9). UML activity diagrams of the initialisation state and the rendering process demonstrate the data flow of the application, depicting the steps involved in the Glut callback mechanism (fig. 5.10).

This approach is simpler in implementation than reading the data in a separate thread or process; however because the sensor and rendering delay are cumulative, it may result in higher system delay. Although, the software could read the sensors and render the scene at the same time, the rendering delay, is much higher than the sensor delay, so such a ‘parallel’ approach

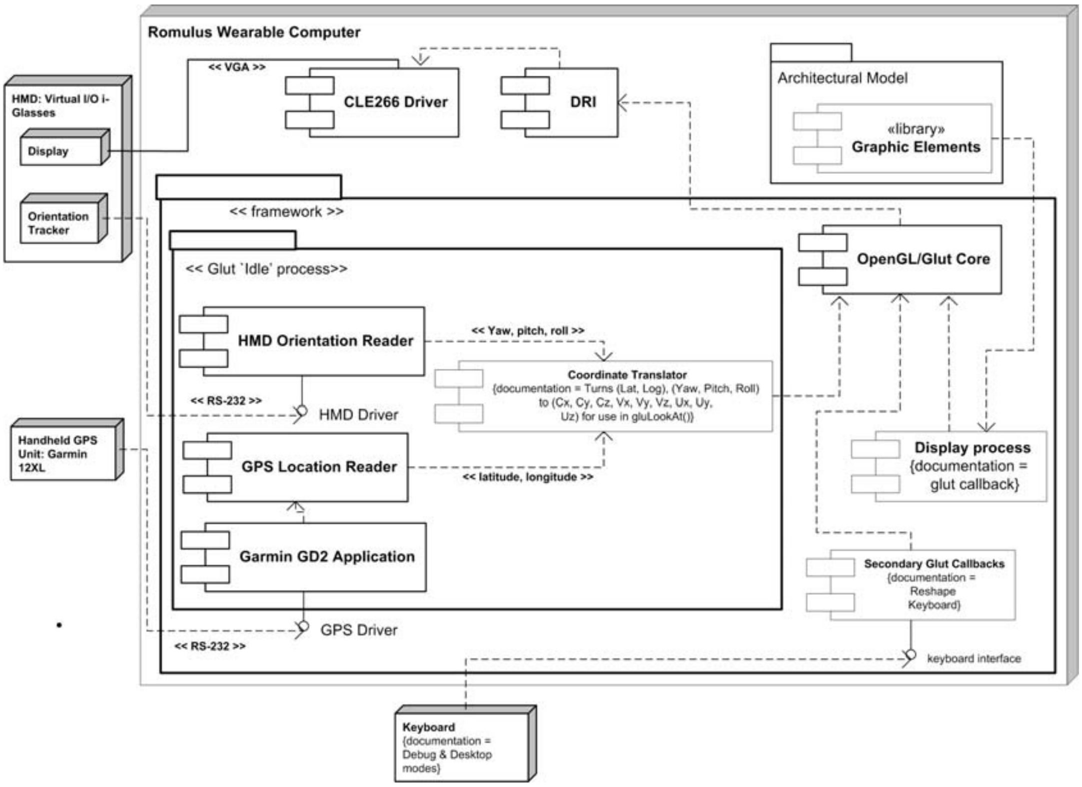


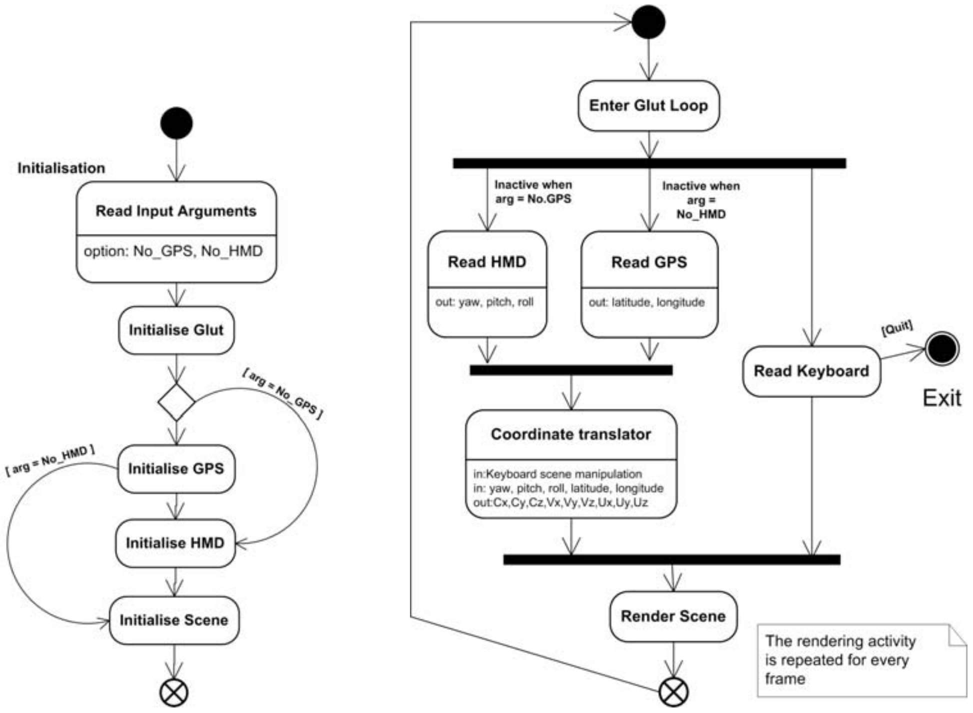
Figure 5.9: Detailed UML deployment diagram of the Gosbecks tour guide application

would not result in significant performance improvement, unless the rendering delay could be reduced. Experience shows that the sequential approach achieves the basic functionality required, yet remains simple and is slightly less demanding of processing power.

The following sections describe the design of the Gosbecks main temple model (Section 5.6) and the two versions of the application implemented during the author's research (Section 5.7 and Section 5.8). The first version achieved basic functionality and served as a development test-bed for the location and orientation determination mechanisms. The second version features improved performance by incorporating a number of speed-up mechanisms.

5.6 The Gosbecks temple model

The application's principal virtual scene is the Gosbecks park main temple complex. Although the main focus of the tour guide, the model remains a separate module of the application and does not form part of the overall architecture. Thus, the application can be used with alternative scenes, simply by replacing this module. The temple architecture was briefly described in Chapter 1; a more detailed description is given below. A detailed architectural design (top view) is included in Appendix A. The reader should note that producing a model of high architectural



(a) Initialisation process

(b) Rendering Process

Figure 5.10: UML activity diagrams of the application's data flow

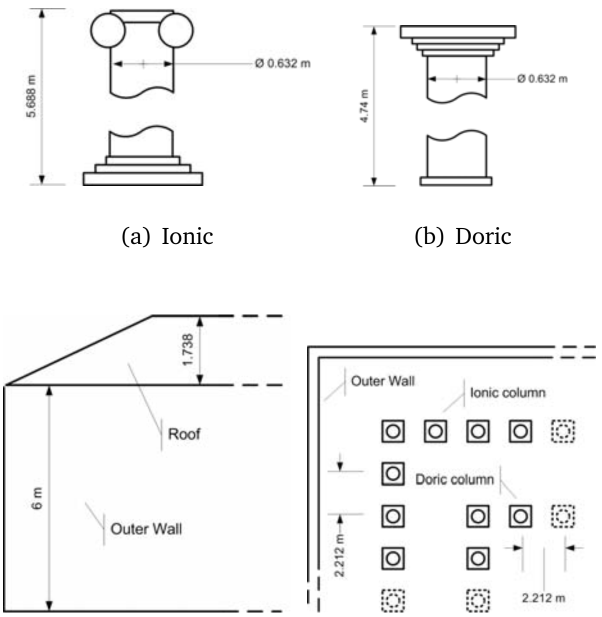
and historical accuracy was beyond the scope of the author's research, though archæologists have remarked that the 3D model seems quite acceptable.

5.6.1 Architecture overview

The temple was surrounded by a square-shaped portico or colonnade [163]. Each side of the portico was separated into sheltered walks, formed between the outer wall and rows of Ionic and Doric columns extending parallel to the temple's sides. A simple, triangular roof covered the portico (fig. 5.11).

The entrance of the temple complex was in the middle of the front side of the portico. A small chamber was formed by surrounding columns, immediately behind the entrance, which had a separate roof vertical to and taller than the portico roof. The entrance was completed by four Ionic columns in the outer wall opening (fig. 5.12).

The interior of the temple complex originally included a ditch extending parallel to the portico, surrounding the main area in the middle of the temple, with a break, behind the temple entrance forming a natural bridge to the main area. The ditch does not exist today and it was eventually replaced by a 2D outline.



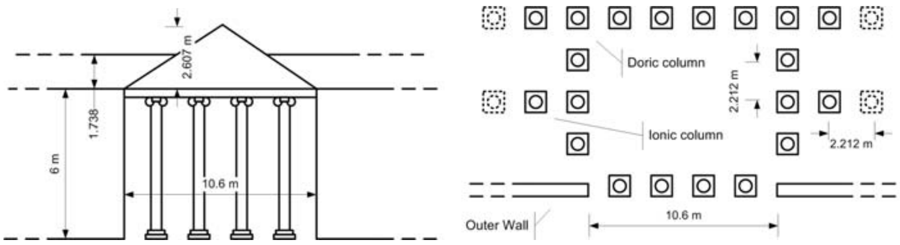
(a) Ionic

(b) Doric

(c) Corner — outer view

(d) Corner — top view

Figure 5.11: Portico details (not in scale)

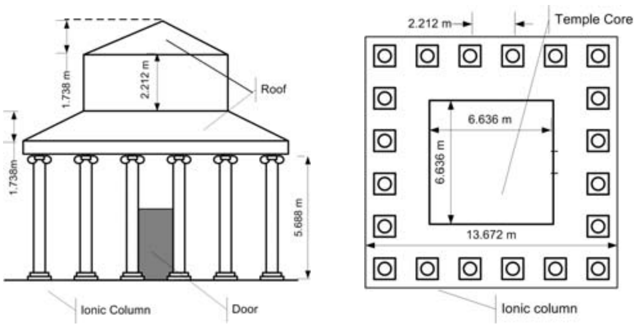


(a) Entrance outer view

(b) Entrance top view

Figure 5.12: The temple's entrance (not in scale)

The main temple was a hexastyle peripteral, meaning it had six columns per side (twenty in total enclosing) the cella and covered by a roof [163]. On the south and north sides of the peripteral all the columns faced forwards. On the east and west, all but the first and last faced forward. The cella's door was on the north side. The peripteral's shape is shown in figure 5.13. Some of the dimensioning uses the paltry remains of the temple and assumes it was constructed in accordance with Vitruvius's architectural principles [151]. Most dimensions are deduced using Vitruvius formulaic rules [151], based on the column diameter (0.632 metres). Any dimensions that were unknown, such as the details of the bases and capitals, were estimated using these rules and the dimensions of similar buildings elsewhere, so that the objects do not look disproportionate. (See also Appendix A for more dimensions.)



(a) Outer View (b) Top View

Figure 5.13: The peripteral (not in scale)

5.6.2 Modelling the temple objects

In order to simplify the construction of the model and focus mainly on dimensioning and correct object placement, a set of primitive shapes was implemented. These were a cylinder, a cube and a pyramid (modifiable to a prism), all with arguments that control their dimensioning and placement (fig. 5.14). These shapes were used as building blocks for the model's objects. All elements of the model were created in a 1:1 scale.

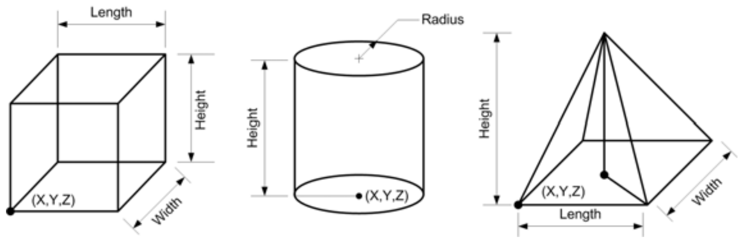
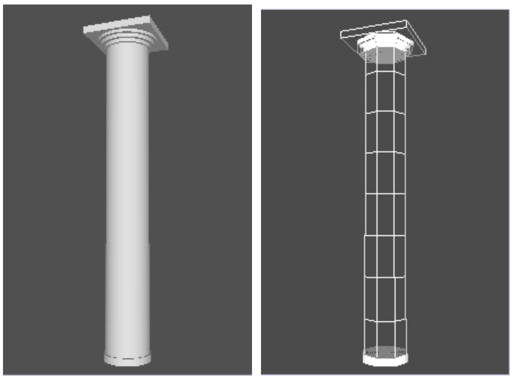


Figure 5.14: The primitive shapes used in the temple model

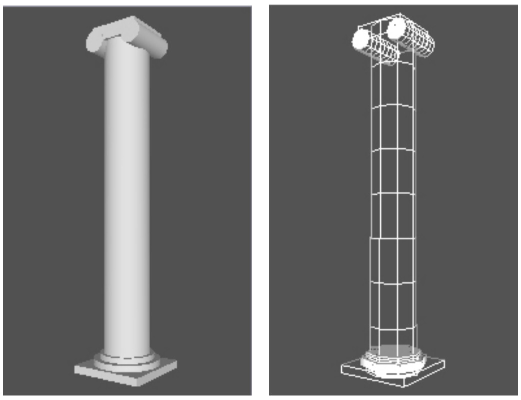
Doric and Ionic columns

Doric columns are fairly simple, made of a cylinder for the shaft, a small cylindrical base and a capital comprising three cylinders concentric to the main cylinder, surmounted by a box (fig. 5.15). The Ionic column is more complex, with a more detailed capital made of two horizontal, parallel cylinders for the volutes with a box between them. The base was made of a box and a pair of concentric cylinders (fig. 5.16). The Ionic columns are not, strictly speaking, accurate as the pediments and bases did not used cylinders; however, we are more concerned with giving a fair reconstruction of the entire Gosbecks site than with the minutiae of the architecture. In principle, such fine detail could be included via the "levels of detail" mechanism expounded below.



(a) Smooth shading (b) Wireframe

Figure 5.15: Doric column models



(a) Smooth shading (b) Wireframe

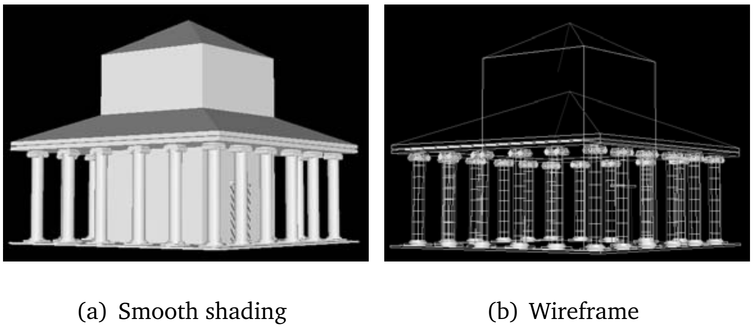
Figure 5.16: Ionic column model

Peripteral

The peripteral was constructed from a set of boxes, forming the base, the cella and the middle-roof epistyle, a marble beam that sits on the columns (fig. 5.17). The roof is essentially a pyramid, cut in the middle from the temple core box. Twenty Ionic columns surround the core of the temple.

Portico and entrance

The Portico is the most complex part of the model, mainly due to the large number of columns involved (fig. 5.18). Each side has thirty-two Ionic columns and twenty-eight Doric. The outer wall, ceiling and the pavement were modelled using the box primitive shape. The roof was modelled using a modified version of the pyramid primitive shape. The shape resembles a



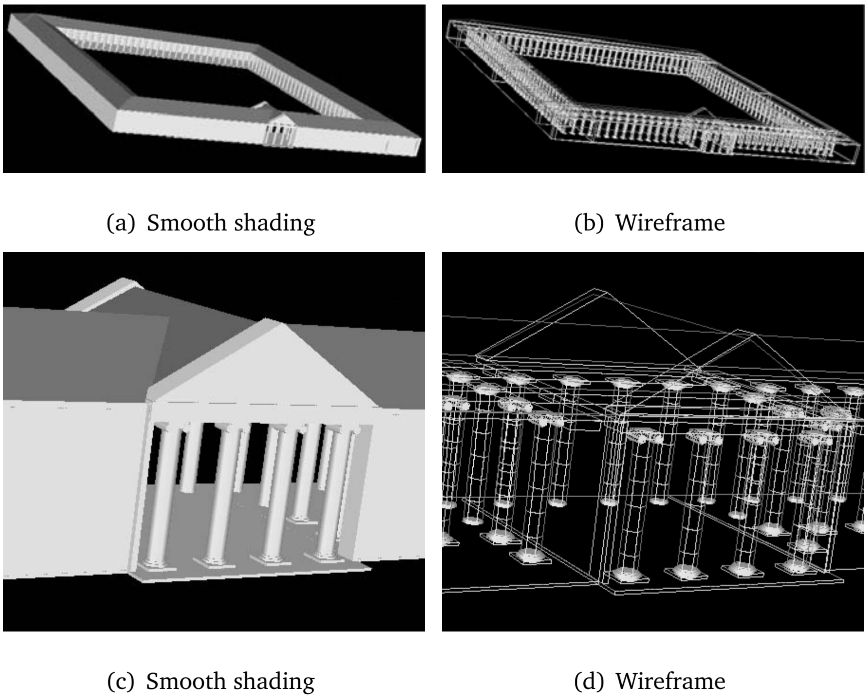
(a) Smooth shading

(b) Wireframe

Figure 5.17: Peripteral model

triangular prism with a spline of variable length[8]. The roof covers all four sides of the portico.

The entrance is rather more complex. Ionic and doric columns existed — following the pattern of the rest of the portico — in all sides of the entrance chamber, whereas the ceiling, pavement and outer walls were modelled using boxes. The entrance roof was modelled from a triangular prism with its spine vertical to the main portico roof. Two more triangular prisms form the detail of the entrance roof (fig. 5.18). The colouring of the temple was kept simple, with the roof brick red and the rest light sand. The ground is a simple green plane.



(a) Smooth shading

(b) Wireframe

(c) Smooth shading

(d) Wireframe

Figure 5.18: Portico model and entrance detail

[8] The spine centre's projection bisects the prism's base in its centre.

5.6.3 Model Specification

The resulting model comprises of 310 objects (columns, walls, pavement etc.) and has four levels of detail. Table 5.1 shows the number of polygons of the model in each level of detail. Normally, the model is used in the lowest level, yielding a higher frame rate. Cylinders, effectively being prismatic primitives with each facet a single polygon, are made to appear realistic by using smooth shading to 'blend' the inter-facet edges. The top and bottom of the cylinders are disks, made out of triangles (slices) equal to the number of the disk's facets. Although the number of polygons used in the model is not high when compared to other 3D architectural reconstructions [6, 73, 247], the model did stretch the real time rendering capabilities of Romulus (Chapter 4). Some techniques to reduce overall polygon count are described in Section 5.8.

Polygon count	
Facets (per cylinder)	Polygons
6	29.907
8	38.727
16	74.007
32	144.567

Table 5.1: Polygon count per level of detail setting

5.7 The tour guide application

The implementation of the tour guide application was based mainly on the aforementioned features of the OpenGL/GLU/GLUT architecture. The application development began in April 2001 with the model design and coding as well as the implementation of a simple framework, demonstrating a simple scene manipulation mechanism using the keyboard. During the autumn of 2001 the software for the GPS and the Virtual i/O HMD tracker were implemented. The first version of the application was tested on Remus during the winter of 2002. Due to poor performance, a number of improvements were identified, along with the need to construct a new wearable platform, Romulus. The application was re-designed with speed improvements such as view frustum culling and the model was re-organised in a scene-graph structure.

5.7.1 Head tracking and view generation

The viewing vector

The gluLookAt() function is used to specify the direction and orientation of view. Data from the head-tracker are used to modify the values of the view point (V_x, V_y, V_z) in such a way that the orientation of the viewer changes the OpenGL camera orientation accordingly.[9]

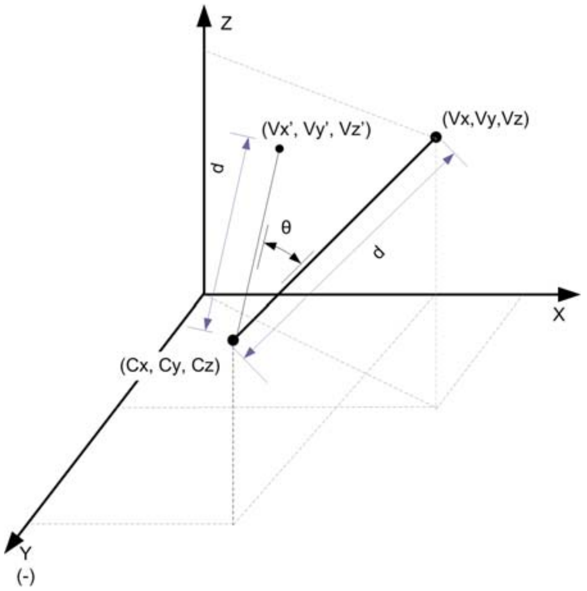


Figure 5.19: The viewing vector

If the distance between the cameras position and the view point points is d and the angle of rotation is θ, the calculation of horizontal rotation (XY plane)[10] is as follows:

$$x_d = V_x - C_x \tag{5.2}$$

$$y_d = V_y - C_y \tag{5.3}$$

$$t = \cos^{-1}\frac{xd}{d} \tag{5.4}$$

if $y_d \geq 0 \Longrightarrow t' = t + \frac{\theta\pi}{180}$

if $y_d < 0 \Longrightarrow t' = 2\pi - t + \frac{\theta\pi}{180}$

Thus the new viewpoint can be deduced from:

$$V_{x'} = C_x + \cos t'd \tag{5.5}$$

[9] Based on tutorials from: http://athene.riv.csu.edu.au/mkavakli/

[10] This calculation changes the HMD coordinate system to the one used in the scene modelling, i.e. Z upwards and Y pointing from the viewer towards the display/page.

$$V_{y'} = C_y + \sin t' d \tag{5.6}$$

Likewise for the vertical rotation (YZ plane): let d be the distance between (Cx, Cy) and (Vx, Vy) and d_2 the distance between (Cx, Cy, Cz) and (Vx, Vy, Vz), therefore

$$x_d = V_x - C_x \tag{5.7}$$

$$x_d = V_y - C_y \tag{5.8}$$

$$z_d = V_z - C_z \tag{5.9}$$

$$t = \cos^{-1}\frac{xd}{d} \tag{5.10}$$

$$t_2 = \cos^{-1}\frac{d}{d_2} \tag{5.11}$$

if $y_d \leq 0 \Longrightarrow t' = \pi - t$

if $z_d \geq 0 \Longrightarrow t'_2 = t_2 + \frac{\theta\pi}{180}$

if $z_d < 0.0 \Longrightarrow t'_2 = 2\pi - t_2 + \frac{\theta\pi}{180}$

Thus the new viewpoint can be deduced from:

$$V_{z'} = C_y + \sin t_2 d_2 \tag{5.12}$$

$$V_{x'} = C_x + (\cos t \cos t_2 d_2) \tag{5.13}$$

The calculated viewpoint coordinates are passed to `gluLookAt()` and, along with the camera coordinates, form the viewing vector of the user. Because the other rotations are implemented using the aforementioned vector, a simple rotation matrix is sufficient, avoiding the effect of gimbal lock [7]. Gimbal lock is a general limitation of rotation matrices and occurs when two rotational axis of an object point in the same direction and causes improper calculation of the viewing vector [83, 174]. It typically occurs when animating an object with rotation represented using Euler angles. In this research due to the limited range of pitch and roll this effect does not occur.

Roll is implemented by rotating the up vector vertically to the viewing vector. If roll is θ and yaw is ϕ then the up vector is calculated from:

$$U_x = -\sin\theta\cos\phi \tag{5.14}$$

$$U_y = \sin\theta\sin\phi \tag{5.15}$$

$$U_z = \cos\theta \tag{5.16}$$

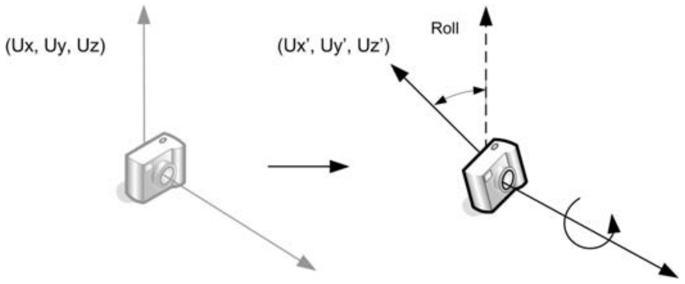


Figure 5.20: 'Roll' by manipulating the 'up' vector vector

Filtering

Initial tests showed a rapid response to head-movement, yet when the HMD remained stable in a fixed orientation there was noticeable jitter. Closer investigation showed that the yaw values fluctuated, in a range of ±3°, whereas pitch and roll remained within ±0.2°. A smoothing function, essentially a running average of the last three values of the yaw the application reads from the tracker, was implemented to stabilise the HMD. The effect is shown in Figure 5.21, where two hundred successive readings, obtained while the HMD remained stable on a glass head model, are plotted with and without filtering. Similar experiments were done in other angles, with the HMD exhibiting the same behaviour. The filtered version, as shown fluctuates less than ±1°. Further assessment of the accuracy of the HMD is provided in Chapter 6.

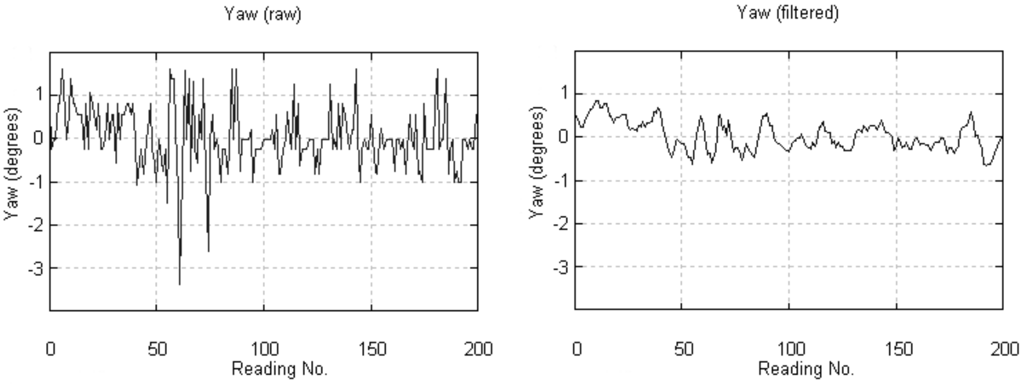


Figure 5.21: Effect of filtering on Yaw using a running average function

Full rotation 'flip' condition

One of the problems encountered after the introduction of the filtering function was related to the operating range of the Virtual I/O HMD. The unit measures yaw angles from $-180°$ to $180°$ (fig. 5.22). When a full rotation is performed, the system detects a transition from $-180°$ to

$180°$. The running average function implemented has an operating window of three values. If the user turns his or her head clockwise a typical sequence of numbers entering the smoothing function could be $179.9°$, $180°$, $-180°$, resulting in an unrealistic average of about $60°$. The result of this problem is a sudden rotation in an orientation opposite to the head movement; subsequently the system returns to the correct direction.

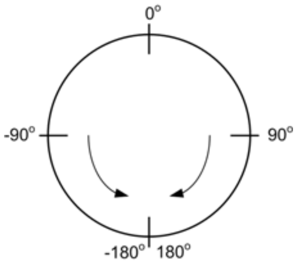


Figure 5.22: The operating range of the HMD

A simple algorithm was implemented to overcome this problem, namely adding or subtracting — depending on the direction of the rotation — $360°$ from the current reading, if a transition between a negative and a positive value is detected. In the example described above, the last (negative) number of the input sequence would become positive and the averaging function would give a realistic result of $180°$.

Applying the aforementioned calculation without any further conditions merely translates the problem to the opposite side, to the origin. The system may detect a transition from $-0.1°$ to $0.1°$, having the same effect as described above. The algorithm therefore must be applied only in the $+90° - -180°/+180° - -90°$ range.

The completed algorithm therefore is:

If $|\theta_i - \theta_{i+1}| > 180° \Longrightarrow$

$$\text{If } \theta_i > 0 \text{ and } \theta_{i+1} < 0 \Longrightarrow \theta_{new} = \theta_{i+1} + 360°$$
$$\text{If } \theta_i < 0 \text{ and } \theta_{i+1} > 0 \Longrightarrow \theta_{new} = \theta_{i+1} - 360°$$

The algorithm yields a smooth transition from $-180°$ to $180°$, filtering the output of the HMD and enabling full horizontal rotation. Pitch and roll are not susceptible to this problem as their operating range is limited to $\pm 60°$.

5.7.2 HMD optical issues

Stereo rendering

As it has been mentioned in Chapter 3, *Stereopsis* (stereoscopic vision) is the perceptual transformation of differences between two monocular images [112], affecting depth perception and is used in binocular displays to provide stereo imagery [102]. Stereopsis, although perhaps not the most important visual cue in depth perception [62], can be controlled in software [95] and add to the notion of immersion by introducing depth to a scene [148].

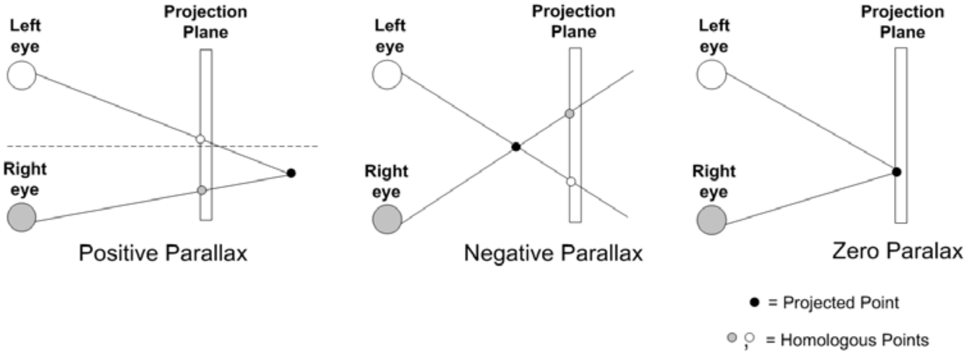
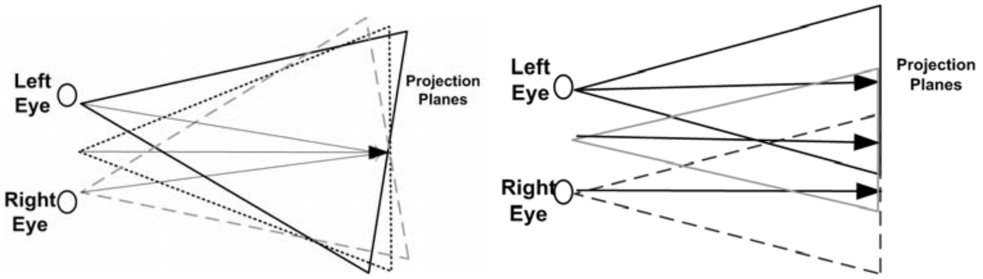


Figure 5.23: Negative, Positive and Zero Parallax

The principle of stereoscopic vision in HMDs is based on the notion of projecting different images on each eye. Stereoscopic depth information is obtained by altering *parallax*, the distance between homologous points on the display screen [62, 85, 95, 148]. Figure 5.23 shows three of the possible values for parallax. A point imaged on the screen has zero parallax, a point behind the screen has positive parallax and a point in front of the screen has negative parallax. When parallax is more than the interpupillary distance it is termed divergent. Vertical Parallax is the vertical distance of homologous points. Both divergent and vertical parallax are undesirable [148].

Two methods have been described for achieving stereopsis on an HMD. The earliest method, denoted rotation or “toe-in” (fig. 5.24(a)), involves the rotation of the object to create a stereoscopic pair of images [148]. Hodges and McAllister [96, 97] describe the problems with this method, principally vertical parallax which is more apparent for objects in the outer field of view. An alternative method, widely accepted as more correct, was introduced by Baker [25] and is the parallel projection, or “off-axis” algorithm (fig. 5.24(b)). The algorithm is based on using parallel image planes, centred opposite each eye, with parallel optical axes [148].



(a) Rotation Method (b) Off=Axis Method

Figure 5.24: Alternative methods for stereopsis on an HMD

OpenGL has the capability to do stereo rendering. In stereo mode, Glut defines two buffers (GL_BACK_RIGHT and GL_BACK_LEFT), the appropriate of which is selected before the rendering operations are performed, using glDrawBuffer(). Unfortunately, this capability is dependent on the driver's capabilities and completeness of Glut support. The versions used in this implementation did not support stereo rendering, therefore such functionality is not available in the application.

However, since the drivers are likely to support Glut stereo rendering in the future, the author provides an example of how it can be implemented. Appendix H includes OpenGL/Glut examples of both the parallel projection and rotation algorithms, based on Paul Burke's tutorials available from:

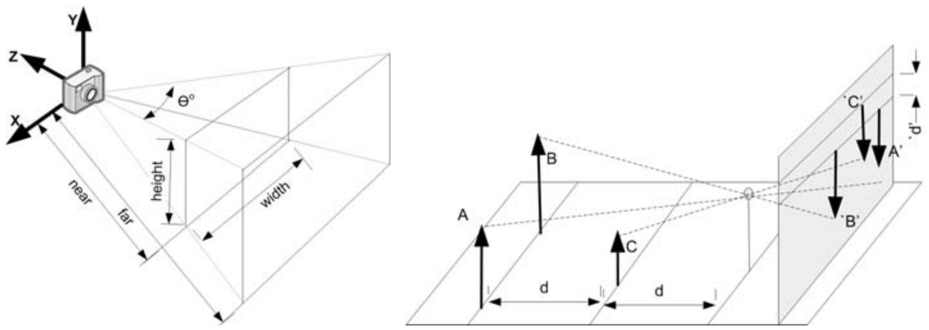
http://astronomy.swin.edu.au/∼pbourke/opengl/stereogl/

Perspective adjustment

The OpenGL camera must be adjusted so that dimensioning and object aspect ratios coincide with those of the real world. OpenGL has provision for perspective adjustment through gluPerspective(), which defines a symmetric view volume:

gluPerspective(fov, w/h, near, far);

where fov is the field of view angle, w/h is the aspect ratio of the viewport and near and far are the near and far clipping planes respectively. In perspective projections, the farther an object is from the viewer, the smaller it appears. This results in a frustum-shaped viewing volume (fig. 5.25). Ideally, the perspective OpenGL camera should match the HMD's field of view [148].



(a) Perspective Frustum (b) Objects appear smaller in distance

Figure 5.25: Perspective projection

The field of view can be calculated by the following equations [148]:

$$\phi = a_{left} + a_{right} - \beta \tag{5.17}$$

$$\beta = \frac{1}{2}(a_{left} - 2\gamma_{left} + a_{right} - 2\gamma_{right}) \tag{5.18}$$

$$a = 2\tan^{-1}\left(\frac{w}{2d}\right) \tag{5.19}$$

where ϕ is the binocular FOV, a is the monocular FOV, β is the angular measurement of the binocular overlap, γ is the angle of each FOV from the forward coordinate axis, w the display's width and d the focal length (fig. 5.26). The Virtual I/O i-glasses have a quoted display image of 80 inches (1.6×1.2 m) at a focal distance of 11 feet (3.35 m). Both monocular displays have FOV axes parallel to the forward coordinate axes. Using the above equations the FOV was calculated at $26°$ approximately, agreeing with the relevant literature [37, 250]. The calculated value was used in `gluPerspective()` along with an aspect ratio of 1.33 (the display uses 640×480 pixel data), with a near plane of 1 m and a far plane of 200 m. The near and far planes were chosen as such so that the complete temple is visible, even when in the distance[11].

5.7.3 HMD calibration

Due to the fact that that the Gosbecks temple no longer has any visible remains, the only reference points on which registration can rely on are the known position of the temple, the archæologist's marks on ground and the horizon. However, Romulus does not include a method for capturing images from the surroundings, therefore no dynamic registering is possible. Furthermore, calibration methods such as the ones described above are tedious to achieve with

[11] The overall temple's length and width are about 85 m (see Section 5.6).

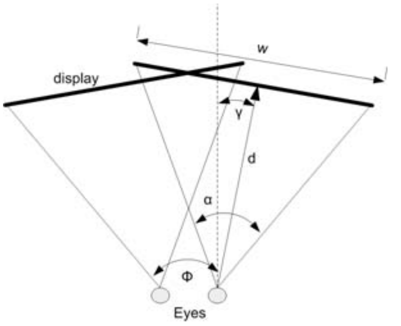


Figure 5.26: Field of view of binocular HMD

inexperienced users, thus reducing the practicality of the system. Moreover, an investigation of HMD calibration algorithms or devising a new one is beyond the scope of this thesis. The author's purpose was to calibrate the HMD as accurately as possible, minimising and preferably avoiding user involvement in HMD calibration, to increase the system's practicality and ease of use. This is actually *aided* by the fact that there is no remaining architecture at Gosbekcs, so any architectural misalignments are not observable.

Bearing in mind that the perspective projection was adjusted as described, the HMD calibration procedure was separated in to three parts; axis alignment, height adjustment and scaling. The immersion in the 3D environment is based on the following assumptions. Provided that the axes of the HMD are properly aligned, the user should start the tour, facing west (the temple faces East) at 14° anti-clockwise with respect to the north – south axis. The viewing vector has to be parallel to the ground, 5 metres away from the Gosbecks entrance[12]. The system uses the initial reading at the starting position to locate the temple relative to the user and calculates the incremental changes from that reference point. This scheme enabled the author to 'place' the temple in alternative positions to aid debugging.

Axis alignment

Arguably, the most important aspect of calibration is the axis alignment, so that the horizon is properly seen through the virtual world. The axes were aligned using a chart-target placed with its origin at eye-height. A similar virtual target at the centre of the focal plane (fig. 5.27) was used to align the HMD to the real target. Since the I-glasses are binocular, calibration methods such as the one described are susceptible to eye dominance effects, so the dominant eye of the user that performs the calibration needs to be used, as pinpointed by Genc *et al.* [79]. It is

[12] These dimensions were measured in reference to the ground markings at Gosbecks.

also assumed that there is no offset between the real and the calculated focal plane centres. Although in reality it may not be so, this kind of accuracy was considered unnecessary in this research.

Using the dominant eye to align the target, the author recorded the values of yaw, pitch and roll, with the real and virtual targets aligned. The process was repeated ten times and the results were averaged. The recorded offsets were used to correct the starting point of the whole process to the origin.

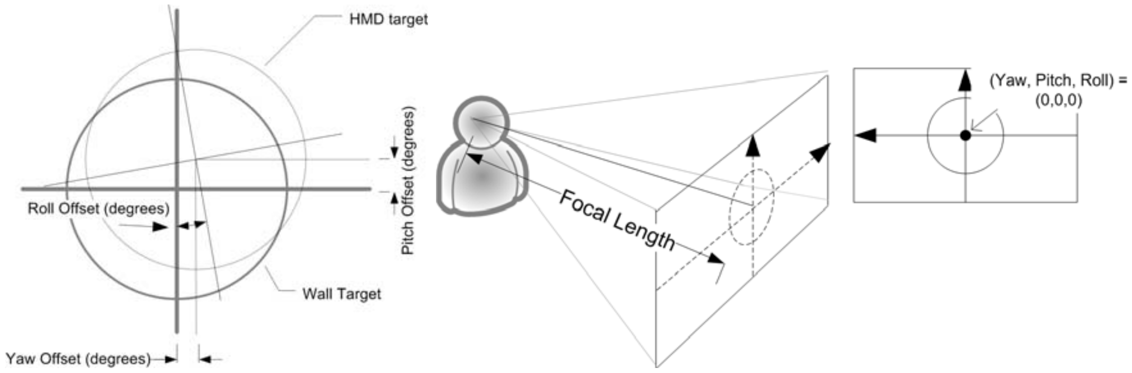


Figure 5.27: Axis alignment

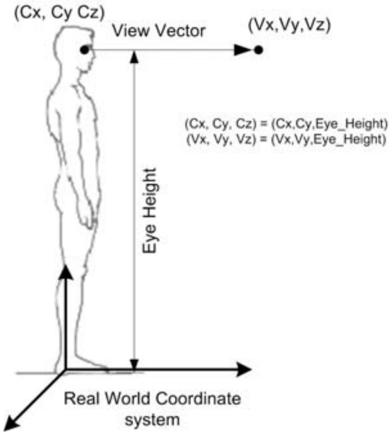

(Cx, Cy Cz)
View Vector
(Vx,Vy,Vz)
(Cx, Cy, Cz) = (Cx,Cy,Eye_Height)
(Vx, Vy, Vz) = (Vx,Vy,Eye_Height)
Eye Height
Real World Coordinate system

Figure 5.28: Height alignment

Height adjustment

Height adjustment, which is also required, poses a further problem. Although the viewing vector in OpenGL has provision for height adjustment (the values of C_Z and V_Z), this can not be deduced dynamically without additional information; in any case, any default value would be wrong for users of different eye height (fig. 5.28). A simple method implemented was to use a keystroke to adjust eye height, however it is considered difficult for users to do so and would require an additional keypad[13]. Currently, height adjustment is a limitation of this system. Most of the aforementioned calibration methods solve this problem by calibrating the HMD for each user, sometimes relying on the expertise of the latter; however in this scenario, this would be impractical. For the assessment described in Chapter 6 the user's eye height was set to be 1.75 metres.

Scaling

Scaling was simpler to determine. Although the perspective projection was adjusted to be the same as the HMD's field of view [148] as described above, and the objects are designed in a $1:1$ scale relative to their real counterparts, discrepancies in object size may occur due to the HMD optics. A simple method to determine the scale of objects was to place a real and a virtual object at the focal distance of the HMD and scale the virtual object until its dimensions agree with its real counterpart. This method assumes that the perspective adjustment is correct and therefore objects at various distances will be scaled appropriately.

5.7.4 The GPS mechanism

As with the head-tracker, the position obtained from the GPS receiver affects the OpenGL camera via `gluLookAt()`, but this time by modifying the camera coordinates (C_x, C_y, C_z). Nonetheless, the calculation is more complicated due to the nature of the GPS coordinate system.

GPS coordinate systems

The main function of the GPS is to supply the application with latitude and longitude,[14] which are then transformed into local coordinates. The Garmin G12XL unit offers three options (Table. 5.2) for outputting information, each with differing accuracy. The Garmin protocol outputs latitude and longitude as radians in a binary double representation but altitude information is not available. The theoretical accuracy is superior to Garmin's implementation of the NMEA protocol. Since altitude is not necessary in this case, the choice of communication protocol was Garmin's proprietary implementation.

[13] A keypad is considered one of the future additions, providing some control of the application to the user, as will be described in Chapter 7.

[14] Altitude is not implemented because the Gosbecks site is flat.

The gd2.c procedure was modified for use with the author's software to output only latitude and longitude conforming to the WGS-84 standard geoid. The World Geodetic System 1984 (WGS-84) is an Earth-fixed global reference framework. Its parameters define the shape of the Earth as an ellipsoid.

Both latitude and longitude are subsequently translated to metres using the Earth Centred, Earth Fixed (ECEF) (fig. 5.29) coordinate system. The origin is at the centre of the Earth and the Z-axis points upwards through the north pole, while the X-axis passes through the intersection of the Greenwich Meridian and the equator.

Communication Protocols — Garmin 12XL	
GARMIN	a binary, proprietary format, capable of downloading/uploading waypoint, track and route data yet performing poorly in other features
NMEA	a standard text-based protocol, used in maritime navigation
TEXT	undocumented and unsupported by Garmin

Table 5.2: Communication protocols offered by the Garmin 12XL receiver

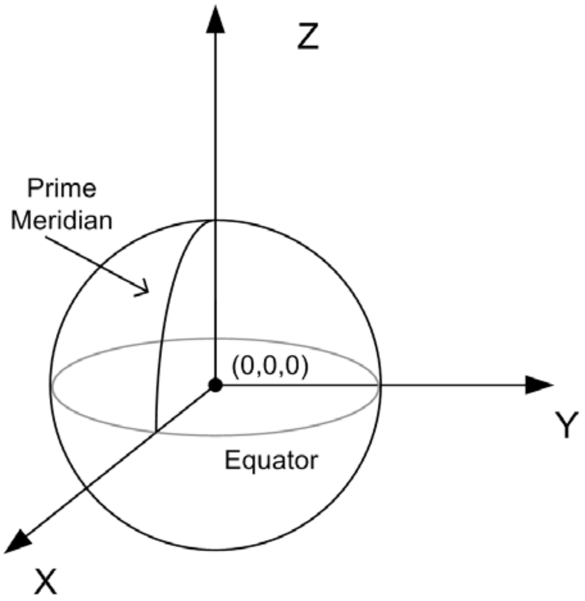


Figure 5.29: Earth Centred, Earth Fixed coordinate system

GPS calculations

Let N be the distance from Earth's centre to the surface,[15] (X', Y', Z') the ECEF coordinates in metres, ϑ the latitude, φ the longitude and η the altitude. Then:

Constants		WGS-84 value
a	semi-major axis	6378137.000
f^{-1}	inverse flattening	298.257223563
$(\delta X, \delta Y, \delta Z)$	datum offset	(0m,0m,0m)

$$b = a \times \frac{f^{-1} - 1}{f^{-1}} \qquad e_1 = \frac{a^2 - b^2}{a^2} \tag{5.20}$$

$$N = \frac{a}{1 - e_1 \sin^2 \vartheta} \tag{5.21}$$

$$X' = (N + \eta) \cos \vartheta \cos \varphi + \delta X \tag{5.22}$$

$$Y' = (N + \eta) \cos \vartheta \sin \varphi + \delta Y \tag{5.23}$$

$$Z' = (N(1 - e_1) + \eta) \sin \vartheta + \delta Z \tag{5.24}$$

The ECEF coordinates have to be translated to local coordinates on the surface of the Earth. The position of the GPS is read and the ECEF coordinates (X_0, Y_0, Z_0) are calculated. A transformation matrix is setup by:

$$r = \sqrt{X_0^2 + Y_0^2 + Z_0^2} \tag{5.25}$$

$$p = \sqrt{X_0^2 + Y_0^2} \tag{5.26}$$

$$\mathbf{E} = \begin{pmatrix} \frac{-y_0}{p} & \frac{x_0}{p} & 0 \\ \frac{-(x_0 \times z_0)}{p \times r} & \frac{-(y_0 \times z_0)}{p \times r} & \frac{p}{r} \\ \frac{-y_0}{p} & \frac{-y_0}{p} & \frac{-z_0}{p} \end{pmatrix} \tag{5.27}$$

where $\mathbf{E}$ is the transformation matrix for transforming ECEF coordinates (X_e, Y_e, Z_e) to Earth surface coordinates:

$$\mathbf{E} \times \begin{pmatrix} X_e \\ Y_e \\ Z_e \end{pmatrix} = \begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} \tag{5.28}$$

The local coordinates (X_c, Y_c, Z_c) are then passed to `gluLookAt()` as the camera coordinates.

[15] adapted from: `http://home-2.worldonline.nl/samsvl/` by David J. Johnston

5.7.5 The tour guide application

The application framework is based on the GLUT callback loop. Initialisation of the scene involves setting up global lighting, some internal model values like the number of facets in columns and coordinates for the positioning of the model.

Temple placement

The 'display' callback draws the temple model. A modelview transformation places the temple and sets up its orientation. The placement of the temple is dependent on the initial GPS reading, acquired as the application is started. As mentioned, the temple is placed facing east with the middle of its entrance facing the user. Placing the temple in an arbitrary position and not 'anchoring' it to the correct latitude and longitude facilitates debugging field tests in the university area. The HMD subsystem on the other hand requires that the temple is positioned in its 'real' orientation (as previously discussed), so that the user's motion in the virtual world has the same orientation as in the real world. A reshape function was implemented to allow for window resizes.

Keyboard support

A keyboard function was also implemented, mainly used for debugging the model and application. The function enables the use of the keyboard for navigation in the temple model, switching between flat and smooth shading, switching between wireframe and solid object rendering and options for the number of column facets (6, 8, 16, 32). The application can be used for desktop demonstration as well, using the keyboard for navigation, by invoking it with appropriate arguments (Table 5.3).

GPS and HMD callback events

The most important stage of the callback loop is the 'idle' function which handles the GPS and HMD. this is invoked during the OpenGL event loop; it reads the GPS and HMD tracker, performs the calculations described above, and modifies the OpenGL camera view by changing the arguments of `gluLookAt()`. The scene is redrawn in the next callback loop iteration.

Main program

The main program has command line options for different operating modes (Table. 5.3). By means of a simple argument the user can choose different input interfaces. Depending on the option, the initialisation of the GPS and HMD mechanisms follows. The temple is also initialised differently depending on the option.

`-nogps`	GPS functionality disabled	Temple placed at predefined point
`-nohmd`	HMD functionality disabled	Temple placement dependent on GPS
`-gpstest`	GPS testing mode	Temple placement dependent on GPS
Default	HMD and GPS enabled	Temple placement dependent on GPS

Table 5.3: Gosbecks command line options

5.7.6 Field tests of the first version of the tour application

Field testing of the application was done using the Remus wearable (see Appendix B) in January 2002 (fig. 5.30). The model was rendered in software via Mesa 3D and without any speed optimisations. These field tests were done principally to assess the functionality of the localisation and orientation mechanisms. Generally, the field tests served as an intermediate development stage and gave the author a better view of the problems inherent in AR. The experience gained aided in the design of Romulus and the improvements of the application, described in Section 5.8.

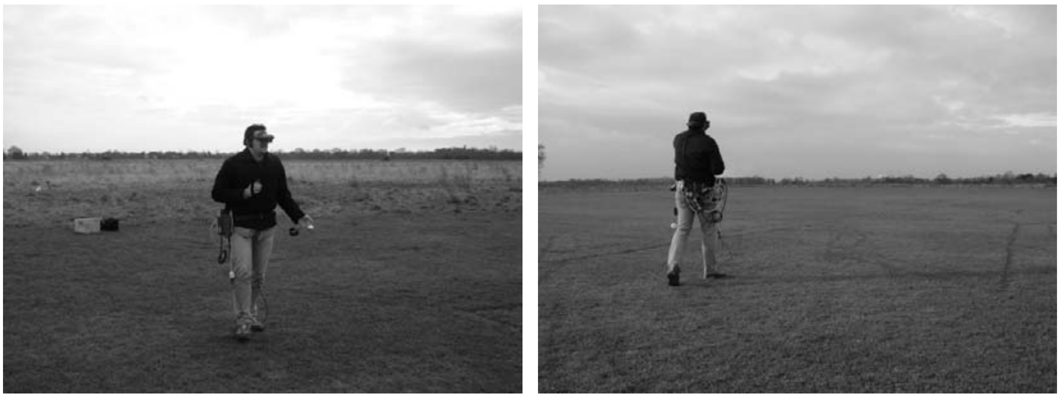


Figure 5.30: Testing Remus and the application at the Gosbecks site

The main bottleneck of the application at this stage was the rendering speed. Remus, a platform that was not particularly powerful, was not aided at all by an un-optimised 3D scene. The drawback was that the full model, which contains more than 29,000 polygons in the lowest level of detail setting (see Section 5.6), was re-drawn with every loop iteration and viewing-vector update. The resulting rendering rate was well below the desired 10 fps, averaging only 0.8 fps. Of course, the portion of the temple that was viewable at any time was much smaller, with only a few objects in the user's view in most cases. Furthermore, due to the resolution of the HMD, details of objects located far from the user, such the capital of Ionic columns, that were not adequately visible were still rendered in full detail. Doric columns comprise of five cylinders and Ionic of six, resulting in 96 and 120 polygons (including base and capital) respectively. However, columns appear as simple cylinders when viewed in distance.

The second version of the application introduced a series of optimisations, such as *view frustum culling* and dynamic *levels of detail* as well as hardware-accelerated rendering. This improved version of the application was used with Romulus in the field tests described in Chapter 6.

5.8 The improved tour guide application

OpenGL itself provides no mechanisms for scene management or rendering optimisations; these have to be implemented separately by programmers in order to improve the speed and realism of the application. The tour guide application involves rendering a relatively complex model (Section 5.6) on a medium performance system (Chapter 4), requiring speed optimisations and scene management in order to achieve the required rendering speed.

One of the simplest, intuitive approaches for speeding up the application is to omit objects that are not visible from the user. This technique is called *view frustum culling* and it is based on the notion of *scene-graphs* [53]. Additional improvements, aiming to reduce the number of rendered polygons even further, control the amount of detail in the rendered objects, depending on their distance from the viewer (or by exploiting architectural features of the temple). Figure 5.31 presents a UML diagram of the second version of the application.

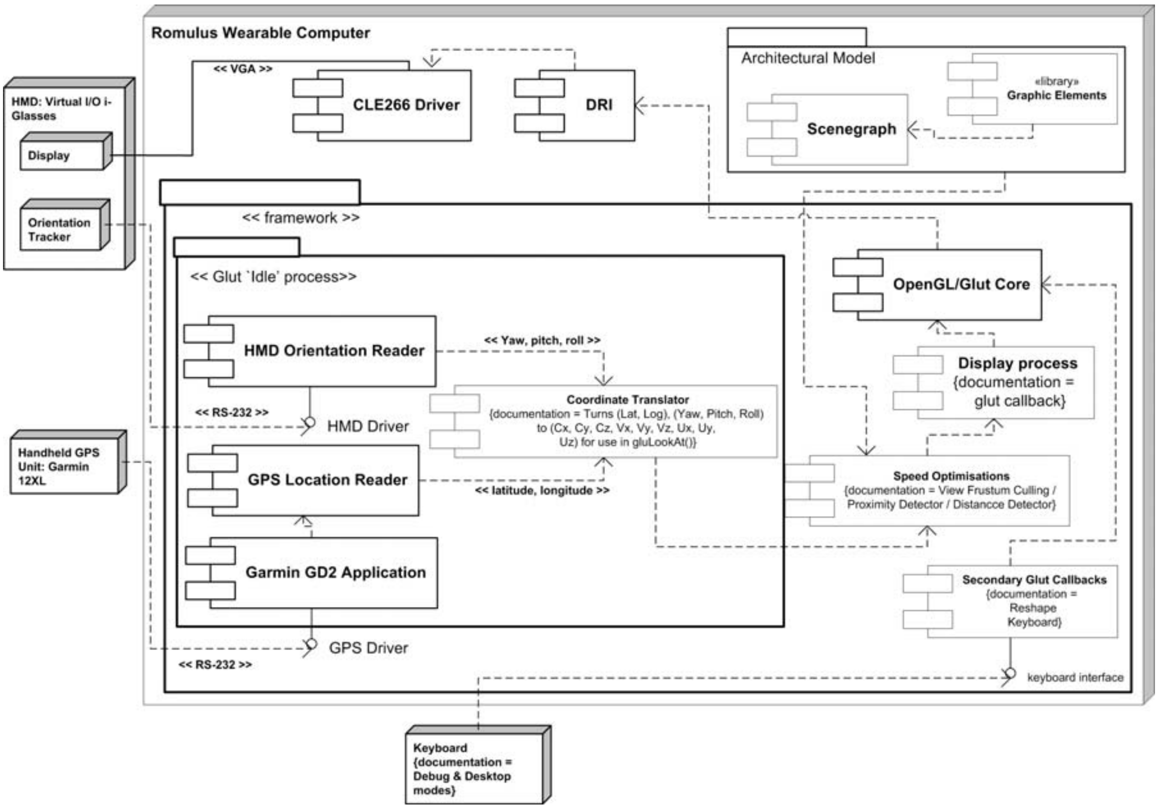

Romulus Wearable Computer
HMD: Virtual I/O i-Glasses
Display
Orientation Tracker
<< VGA >>
CLE266 Driver
DRI
Architectural Model
«library» Graphic Elements
Scenegraph
<< framework >>
<< Glut `Idle' process>>
OpenGL/Glut Core
HMD Orientation Reader
<< Yaw, pitch, roll >>
Display process {documentation = glut callback}
Coordinate Translator {documentation = Turns (Lat, Log), (Yaw, Pitch, Roll) to (Cx, Cy, Cz, Vx, Vy, Vz, Ux, Uy, Uz) for use in gluLookAt()}
<< RS-232 >>
HMD Driver
Handheld GPS Unit: Garmin 12XL
GPS Location Reader
<< latitude, longitude >>
Speed Optimisations {documentation = View Frustum Culling / Proximity Detector / Distancce Detector}
Garmin GD2 Application
Secondary Glut Callbacks {documentation = Reshape Keyboard}
GPS Driver
<< RS-232 >>
keyboard interface
Keyboard {documentation = Debug & Desktop modes}

Figure 5.31: UML Deployment diagram of second (optimised) version of the application

5.8.1 Scene-graphs

Unlike the first version of the application, which had no hierarchy of virtual objects — all elements are rendered, as they are considered of equal importance — 3D scenes can be arranged in a scene-graph, a hierarchical structure that separates elements into spatial groups (fig. 5.32). Scene-graphs are tree-like structures made of nodes and 'leaves'. The top node is the complete 3D scene, the 'world'. The next series of nodes are the 'children', in principle large subsets of sub-elements. Going down the hierarchy, objects are grouped in nodes based on their positions in the virtual world.

Organising the scene into groups allows selective rendering, reducing the number of vertices required per scene redraw. By rendering fewer vertices, a considerable speed-up can be achieved in complex scenes. The objects that are not visible are *culled*. Culling is the process ignoring objects that are not visible in the image of the scene, including things that are behind the viewer, off-screen or, in more advanced systems, hidden behind other objects.

The tree structure speeds up considerably the process of checking the visibility of each node in a complex scene. If a node representing a spatial group is visible, the visibility algorithm tests

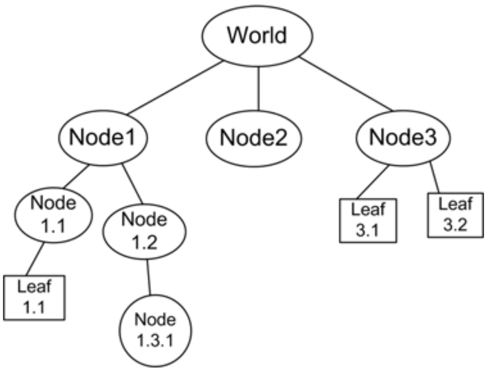


Figure 5.32: An example of a scene-graph

each of the node's children individually, deciding if they are to be rendered or not. On the other hand, if a node is not visible then the tests for its children are not required. If a scene is sensibly fragmented spatially, the hierarchy is an efficient way of scanning the scene and deciding which elements are visible and which are to be culled. The benefit of this technique lies in the fact that performing the required calculations is much faster than actually rendering those objects. The decision of what a user sees involves computations against the viewable volume, the *view frustum*.

5.8.2 View frustum culling

In view frustum culling, each object's spatial boundaries are compared to the viewing frustum. The viewing frustum is a truncated pyramid that represents the visible volume of the user/camera (fig. 5.33). In a perspective projection, the viewing frustum is made out of six intersecting planes, two of which are parallel, the far and near clipping planes.

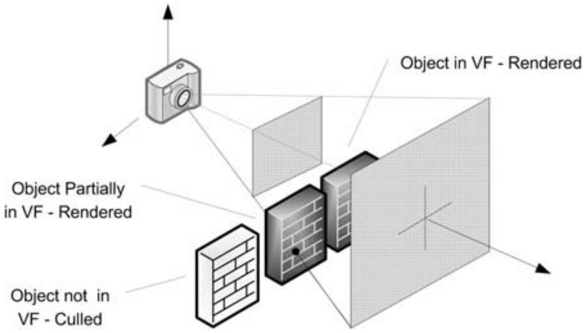


Figure 5.33: View frustum culling

Testing if an object lies within the view frustum requires some geometrical calculations. Each plane of the view frustum has an equation of the form:

$$Ax + By + Cz + D = 0 \tag{5.29}$$

Let an arbitrary point be (x_p, y_p, z_p). The distance of this point from each plane has the form:

$$d = Ax_p + By_p + Cz_p + D \tag{5.30}$$

The distance of a point from a plane will indicate on which side of that plane the point is. Assuming a right-handed coordinate system is used and considering, for example, a horizontal plane, if the distance is positive the point will be above the plane. If it is negative it will be below the plane. If the distance is zero then the point lies on the plane. These three conditions are the basis of the view frustum culling checking algorithm indicating that a point is inside, outside, or on the edge of the frustum. For this to be accomplished the point must be tested against all planes.

The view frustum culling algorithm is implemented in OpenGL by concatenating the Modelview and the Projection matrices. This is performed once per frame. From the resulting clipping matrix, the view frustum plane equations can be deduced [246].[16].

$$\begin{pmatrix} Projection \\ Matrix \end{pmatrix} \times \begin{pmatrix} Modelview \\ Matrix \end{pmatrix} = \begin{pmatrix} Clipping \\ Matrix \end{pmatrix} \tag{5.31}$$

Bounding volumes

Although the above description deals with a single point, the algorithm can be applied for any number of points. However, in practice this can be computationally intensive and may degrade performance when the objects to be tested are complex, with a large number of vertices. A mechanism to overcome this is the bounding volume (BV).

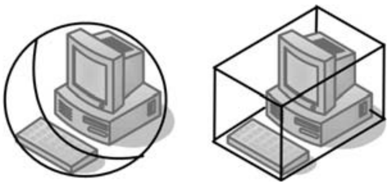


Figure 5.34: Bounding volumes of a sphere and a box

[16] http://www.markmorley.com/opengl/frustumculling.html

A bounding volume is a coarse volumetric description of each object. It surrounds the object in such a way that no object vertices are outside it; it 'encapsulates' the vertices. There are two types of bounding volumes in common use, a box and a sphere (fig. 5.34). In the application described here, a box is used due to the nature of the objects — long columns— that are to be rendered. Using the bounding volume in the tests against the view frustum simplifies the process when objects are complicated and assembled from many vertices.

The view frustum culling algorithm tests whether all 8 vertices of the bounding volume of an object are inside the view frustum. If any is, the object is visible, wholly or partially. If it is a node, the test is repeated for its sub-objects. If the object's bounding volume is entirely outside the view frustum, it is not rendered (fig. 5.33).

5.8.3 The Gosbecks scene-graph

The efficiency of view frustum culling is based on the hierarchical testing of objects in the scene. Fragmenting the scene has to be done in such a way that the scene does not 'break' i.e. there are no positions where objects are culled where they should not.

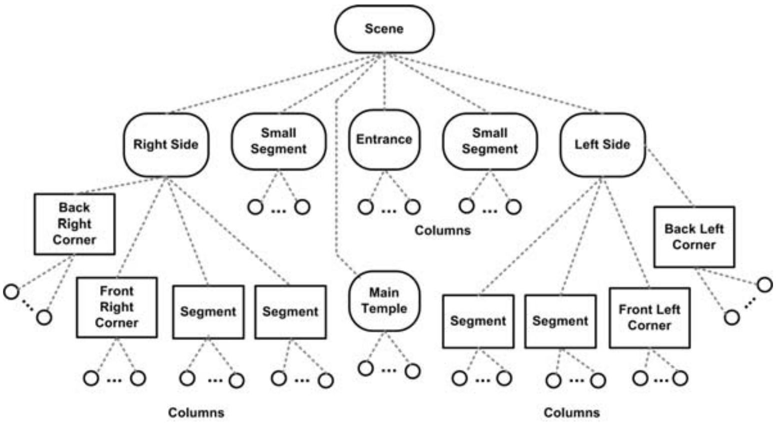


Figure 5.35: Gosbecks temple scene-graph

The Gosbecks temple was separated into 14 sections: the entrance, the peripteral, four portico corners, two portico segments adjacent to the door and six large portico segments. Each section is a 'node'. Some spatially adjacent nodes form 'groups'. The left and right sides (as the viewer looks at the entrance from outside) form two groups, comprising two portico corners and two portico-segment nodes. The rest of the portico-segment nodes are not part of larger groups. The last node is the peripteral. Each node has its own bounding volume. All nodes have walls and columns as their 'children'.

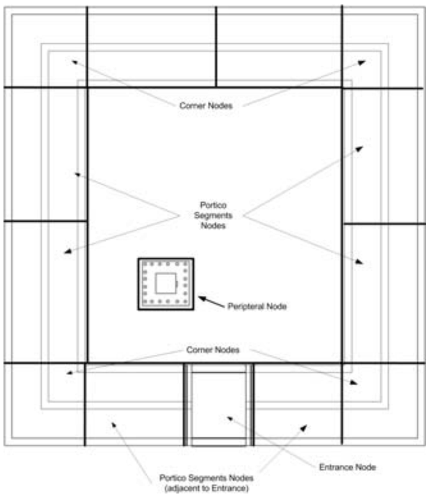
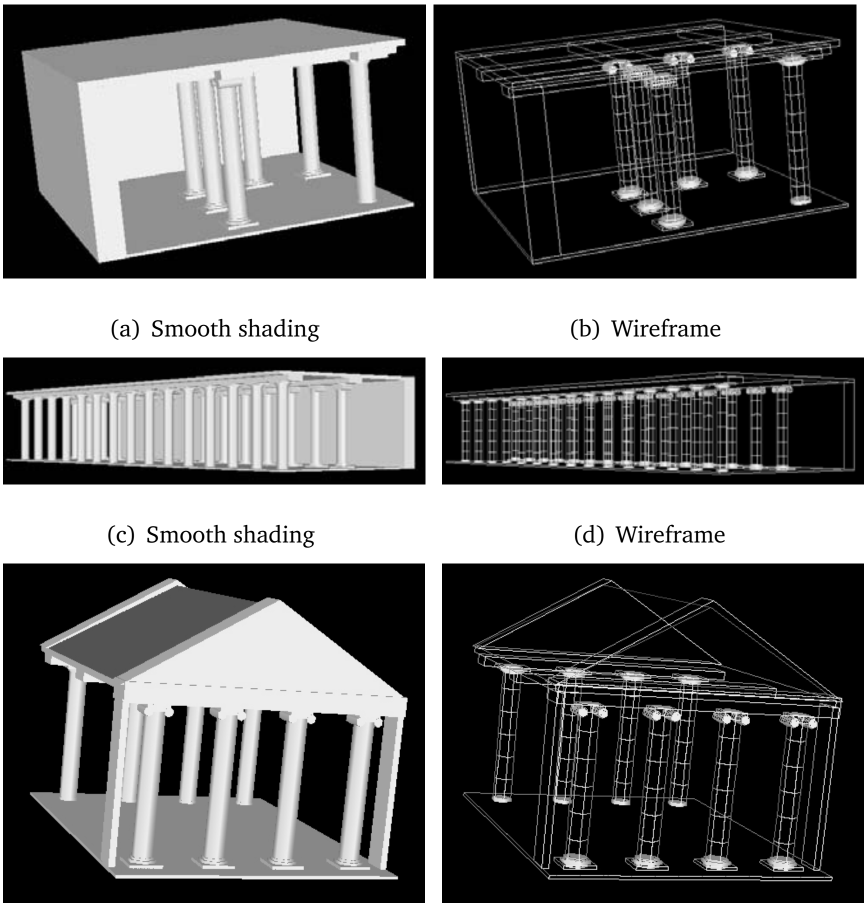


Figure 5.36: Gosbecks temple fragmentation



(a) Smooth shading

(b) Wireframe

(c) Smooth shading

(d) Wireframe

(e) Smooth shading

(f) Wireframe

Figure 5.37: The corner, portico segment and entrance nodes

5.8.4 Further optimisations

In addition to view frustum culling, two further speed optimisations were implemented: a distance detector and a proximity detector, both controlling the detail with which the columns are rendered, depending on their distance from the viewer. These implementations exploit architectural features of the temple. Both speed optimisations become more effective due to the limited resolution of the Virtual I/O HMD and the effects of perspective, where objects in the distance present little detail.

Distance detector

In the Gosbecks temple, there are two rows of columns in each portico side and the front row usually obscures the back row when the viewer is at a distance. This can be exploited by removing the back row of columns if the user is far from the portico side (fig. 5.38). This mechanism works only on the straight segments and not at the corners, where the columns always need to be visible when the user walks in the portico, since one column in the middle of the sheltered walk end is always visible.

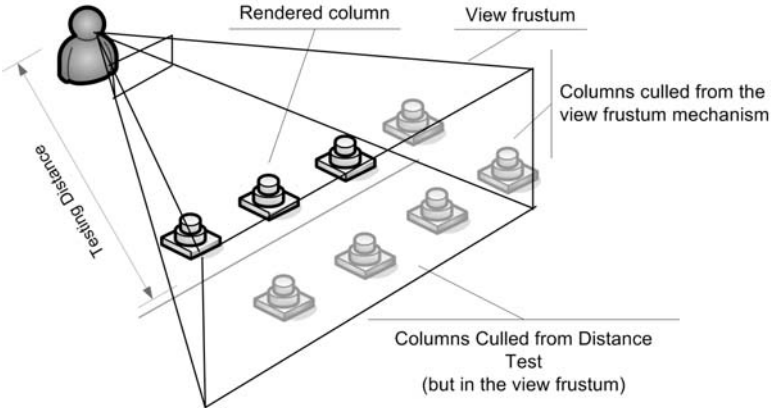

Rendered column
View frustum
Columns culled from the view frustum mechanism
Testing Distance
Columns Culled from Distance Test (but in the view frustum)

Figure 5.38: Distance test

The number of polygons that are rendered when distant is significantly reduced by this scheme. If the user is well away from the sides of the portico, such that many columns are in the view frustum, more polygons are rendered. The distance detector balances to some extent this problem by reducing the number of rendered columns and therefore polygons. On the other hand, when the user is closer to the sides, both column rows are rendered; yet the number of rendered columns is reduced at the same time by the view frustum culling mechanism. Hence, the to speed optimisations are complementary.

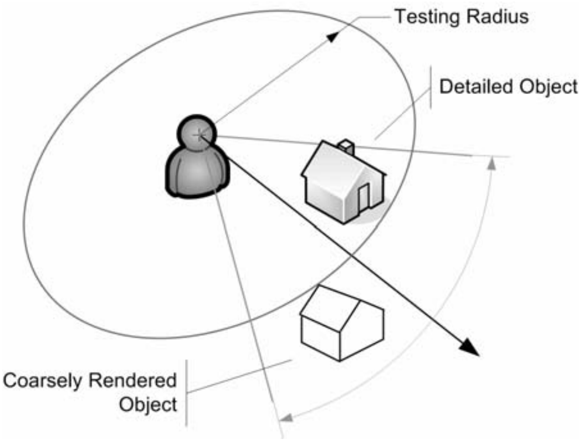


Figure 5.39: Proximity test

Proximity detector

In a similar fashion to the distance detector, the proximity detector reduces the amount of detail of the visible columns, i.e. those that are not culled by view frustum culling and the distance test, by simplifying their geometry (fig. 5.39). Bearing in mind that the columns are constructed from 5 or 6 cylinders, of which one is the shaft and the rest form the capital and base, reducing the detail of each column to a single cylinder when they are far from the viewer reduces even further the number of polygons to be rendered. This mechanism uses a circle as the testing boundary. The circle's radius is the testing distance. If a column is within that radius it is rendered in full detail. If it is outside of the boundary, it is rendered as a simple cylinder.

5.9 Performance gains of the improved tour guide

The optimised version of the application, using the Gosbecks model described in Section 5.6 at the lowest detail setting (29,907 polygons), was tested on Romulus with hardware rendering enabled and the location and orientation determination mechanisms operational.

The increase in frame rate was significant. A visual frames-per-second counter was implemented in OpenGL in order to provide feedback on the rendering speed of the application. This counter essentially measures the system's period, i.e. the time required per iteration loop and hence encompasses the deduction of orientation and location from the HMD and GPS. The performance improvements are summarised in Table 5.4 where the average frame rate is shown, for various configurations and optimisations.

Optimisation Stage		FPS
Remus with tour V.1	No optimisations	0.8 fps
Romulus with tour V.2— no DRI	No optimisations	4.3 fps
Romulus with tour V.2—DRI	No optimisations	7.4 fps
Romulus with tour V.2—DRI	View Frustum Culling (VFC)	9.2 fps
Romulus with tour V.2—DRI	VFC and Distance Test	11.3 fps
Romulus with tour V.2-DRI	VFC, Distance and Proximity Test	12.7 fps

Table 5.4: Performance gain from optimisations

To demonstrate the effect the optimisations had on reducing the complexity of the scene, and therefore improving rendering speed, the author recorded the number of rendered polygons per frame for 2000 frames. The result is shown in Figure 5.40. The number of rendered polygons, is much lower than the total of 29,907 for the complete scene, averaging a 1202.5 polygons per frame (s.d. = 546.22). However, it must be noted that the number and type of objects, and therefore polygons, that are being rendered is directly dependant on the user's movement in the virtual environment.

Due to the fact that the number of polygons rendered is relative to the user's position and view in the virtual world, the frame rate fluctuates. Watson *et al.* [242] claim that in systems where a minimum frame rate is met (10 fps or 100 ms system delay), small variations of the latter do not affect significantly task performance. However, they also state that as variations become more severe, user performance is affected and is dependent on both the amplitude and the period of fluctuation.

Figure.5.41 presents the variation in frame rate over a period of 120 seconds. The average frame rate of this sample is 12.74 fps (s.d = 2.39). It is apparent that the variations are fairly large, yet the system rarely drops below 10 fps, whereas in some cases it reaches a maximum of about 18 fps. These peaks occur when looking towards the corners of the portico, when in close distance, where the number of objects rendered is reduced. The lowest values occur when entering the temple, looking towards the peripteral from distance and when looking inwards from the temple corners, where many more objects have to be rendered. The experiments were repeated ten times and in all cases the system exhibited similar performance.

Although the overall frame rate is far less than the rendering speed encountered in modern games, it is adequate as a rendering speed of 10 fps is considered an acceptable minimum in virtual environments [138, 242]. This minimum has been met, despite the constraints of a large model and a low-power, low-weight, low-cost platform.

As the relevant literature focuses on VE and not in AR, there may be differences on what a user considers acceptable in a wearable AR. Bearing in mind that in AR, the view of the real world works as a reference and that human vision is susceptible to small discrepancies, frame rates of 10—15 fps may be restrictive. For that reason, the system's performance was evaluated from users, as will be described in Chapter 6.

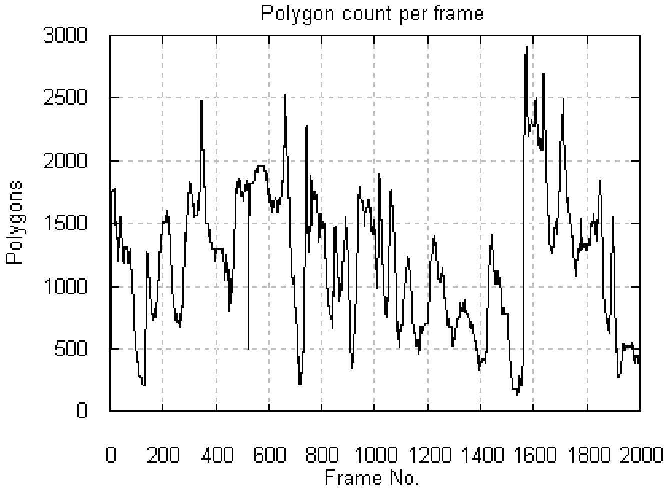


Figure 5.40: 'Real-time' rendered polygon count

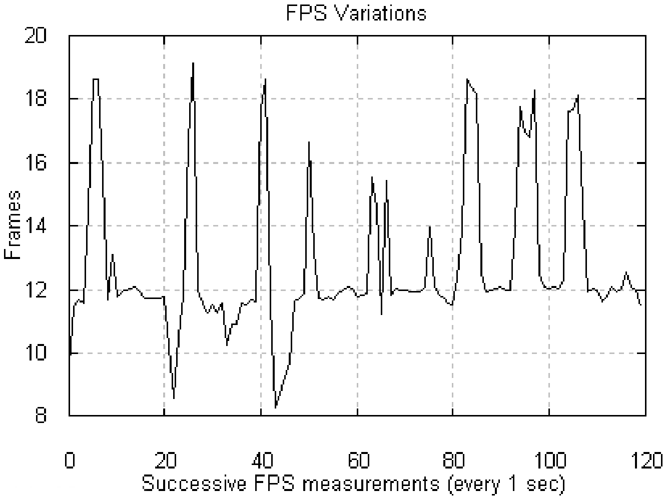


Figure 5.41: Frame rate fluctuations

5.10 User's viewpoint images

The following images depict the user's viewpoint, looking through the HMD while the application is running.

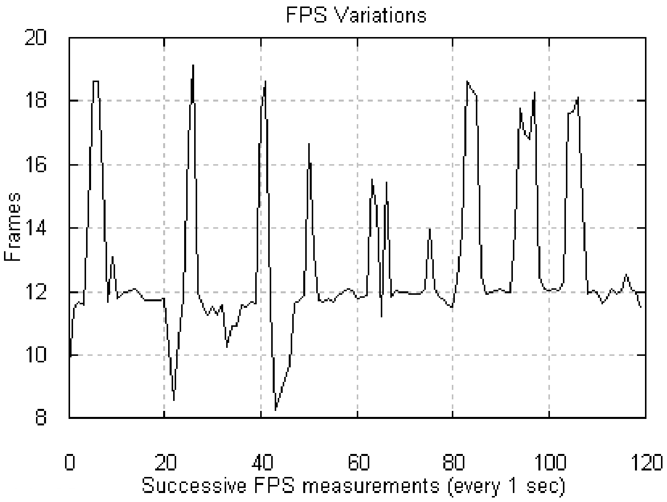


Figure 5.42: Images from the User's viewpoint

5.11 Chapter summary

This chapter described the design and implementation of an augmented reality framework, written in OpenGL. The tour guide application has been designed to function as a demonstration for wearable AR, employing 3D architectural reconstructions, in order to provide an *in situ* guide to the temple complex of the Gosbecks archæological park.

The application is separated into different subsystems. The first that was implemented was the model of the temple, based on archeological information and drawings. User position is derived from a handheld GPS unit connected to a serial port. The location measurement subsystem extracts latitude and longitude from the unit. The orientation measurement subsystem uses a head-tracker to derive information on the user's view direction, also via a serial port. Information from both sensors is translated to the local coordinate system and used to control the OpenGL's virtual camera. As the user roams around the Gosbecks site, the rendered view of the virtual model is updated in near-real time.

Various optimisations were used in order to achieve a frame rate over 10 fps. The temple model, consisting of 29,907 polygons in the lowest detail setting, was modelled using a scene-graph, a hierarchical structure of spatially-arranged objects. View frustum culling and two independent levels-of-detail mechanisms were used to reduce the number of polygons rendered per redraw to an average of about 1200 polygons per frame and hence increase the application's frame rate to a maximum mean speed of 12.7 frames per second. The performance of the application was also aided by the use of hardware-accelerated rendering.

The Gosbecks tour guide and the Romulus wearable computer were intended to investigate whether it is possible realise a demonstrable system that is small, light, and easy enough to use, has adequate performance, realism and accuracy while maintaining simplicity, low-cost and simple hardware–software interfaces. The use of a real archæological building complex as a main theme of the application imposes certain restrictions in terms of realism requirements and rendering complexity. These requirements are likely to be encountered in commercial solutions, if these ever become available.

Chapter 6 that follows, presents a series of quantitative experiments carried out to investigate the accuracy and stability of the sensory sub-systems. It also presents a questionnaire-based evaluation aiming to assess the usability of Romulus and the Gosbecks tour guide and their potential for further outdoors-AR research and demonstrations.

Chapter 6

Accuracy Testing and User Assessment

6.1 Introduction

As has been mentioned, few research efforts in outdoor wearable AR have reached a satisfactory level in terms of functionality. In the author's opinion, the Touring Machine [71] and Tinmith-Metro [167] are probably the most important and successful examples of outdoor, wearable, untethered AR that employ 3D architectural reconstructions.

Although much research has been conducted in the context of sensory modalities, particularly on HMDs and orientation trackers, a practical appraisal of how wearable systems in scenarios such as the one described here are perceived by users is still lacking. The reason is that the technological constraints imposed on most wearable AR systems in terms of performance have led most researchers to concentrate principally on 'proving concepts' or using textual or non-visual interfaces. The fact is that wearable AR systems are susceptible to a series of problems such as low processing power, lack of 3D hardware acceleration and low battery life.

Generally, AR implementations suffer from a series of problems, such as improper registration, inadequate interface design and various perceptual issues. The use of wearable computers in AR scenarios introduces even further problems, discussed in Chapter 3. Further investigation of AR applications using wearable computers requires that the hardware is of adequate graphic processing power, easily constructed, duplicated, maintained and deployed in real experimental conditions, i.e. *in situ*. Romulus, along with the described application framework, aims to do that by augmenting an example 3D reconstruction, the Gosbecks main temple.

The following sections describe a series of laboratory and in-the-field experiments, conducted with the Romulus wearable computer running the Gosbecks tour guide application. The aim of the first series of experiments was to assess the positional and orientational accuracy of the system. The second part of the investigation, a usability survey, aimed to gauge user opinion on the wearable's ergonomics and the system's overall accuracy, visual output quality and sense of presence — effectively an effort to understand how users would perceive this type of application. By identifying what users might perceive as good quality, responsive, accurate, and easy to use, we can learn how research prototypes can be made more practical.

The ultimate aim of this research is to explore how to build wearable AR systems that do not merely prove the concept of AR but can be used successfully in *in situ* practical, experimental investigations. This will in turn aid further research on wearable AR.

Section 6.2, Section 6.3 and Section 6.4 that follow deal with experimental investigation, assessing the accuracy and stability of the GPS and HMD trackers of Romulus while running the wearable AR application framework described in Chapter 5. Section 6.5 and onwards describe the user assessment of the system's usability.

6.2 Assessment of localisation and orientation accuracy

The main challenge of AR applications is to register correctly the real environment with virtual objects and the user's movements. The tour guide is an example of such an AR application with a few key characteristics. Firstly, there are no remains of the real temple above ground on the Gosbecks site and hence no architectural details to register the virtual model to. In its present state, Gosbecks has only ground marks that define the outline of the temple. Secondly, the area is protected by English Heritage[1] which does not allow any equipment to disturb the ground, such as radio masts and antennaes.

Registration therefore relies purely on the GPS input. The user starts the tour guide in front of the entrance to the temple (see Chapter 5). The model is then placed, according to the initial reading from the GPS, in front of the user so that it 'sits' properly on the ground marks. Temple placement is not dependant on any other feature. As the user roams in the virtual environment, the temple ideally needs to be stable in terms of its original real-world position. Moreover, the distance travelled in both worlds needs to be the same: if the user walks for thirty metres in the real world, exactly the same movement needs to be performed in the virtual world.

The requirement is to have accurate, real-time positioning. As mentioned in Chapter 1 the wish that the user should perceive the 3D reconstruction as being 'in the right place' implies an

[1] http://www.english-heritage.org.uk/

accuracy requirement for position of the order of 1 cm. A more achievable target for wearable AR is that doorways in a 3D reconstruction are sufficiently stable positionally that the wearer can negotiate physical doorways without difficulty, say less than 0.1 m.

However, most current wearable AR systems achieve accuracies of the order of 1–2 m [71] normally and in ideal situations 50 cm. Piekarsky *et al.* [166] and Avery *et al.* [16] claim the highest positional accuracy of a wearable computer system as 50 cm, achieved from their Tinmith wearable equipped with a Trimble Ag132 GPS unit with DGPS functionality. They also state that this is achievable under favourable conditions and that positional accuracy of the order 5 m is more often achieved [16]. As stated in Chapter 5 the aimed accuracy of this research system is 2 m.

Furthermore, the HMD tracker needs to be able to detect accurately the user's head movements. Accuracy requirements for orientation should be less than 1° error. However, as discussed in Chapter 1, in order to keep errors to a fraction of a degree ($<0.5^\circ$), end-to-end system delay should be kept to 10 ms [18, 21], a performance figure that could be technically difficult for wearable computers. It must also be noted that these target figures may be high enough to induce motion sickness effects in extreme situations [74, 183, 184]; however in the context of tour guides they may be sensible[2]. The system described here has a claimed orientation error of less than 1°.

6.2.1 Statistical tools

All the data collected in the laboratory and field assessments were analysed using a two-tailed t-test [54] with the null and alternative hypotheses as follows:

$$H_0 : \mu_0 = k \rightsquigarrow H_1 : \mu_0 \neq k$$

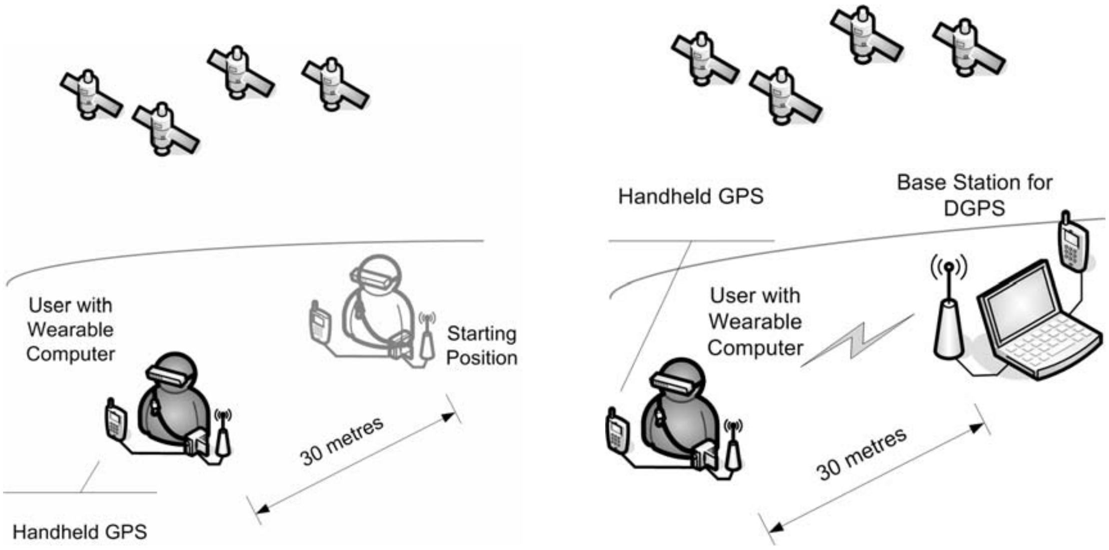
where μ_0 is the hypothesised mean and k an expected value. The expected value in all tests is the one that would be obtained if the orientation and localisation sub-systems behaved with the utmost accuracy. The t-test allows one to assess whether the data tend to give the expected values or not, even though the results are neither exhaustive nor absolute. Furthermore, the real-time performance of each sub-system is depicted graphically for each test.

[2] The user assessment described in Section 6.5 attempts to investigate whether the above figures introduce any undesirable symptoms such as dizziness, disorientation etc.

6.3 Measuring the accuracy of GPS and Differential GPS

6.3.1 Experimental setup

The tests were conducted at the Essex campus for both standard and differential GPS (fig. 6.1). Both experiments were performed using a laptop and the Romulus wearable computer. For the positional stability tests only the wearable was used, remaining stationary while it detected its position every two seconds, for time periods of more than ten minutes. The positions were logged in text files and subsequently analysed.[3] The second experiment evaluated the capability of GPS to detect displacements. The wearable was used to detect its position while walking for a distance of 30 m from a starting position measured at the beginning of the experiment (fig. 6.1.a). The total measured distance was then compared to the real value of 30 m.



(a) Standard GPS Experiment

(b) Homegrown DGPS Experiment

Figure 6.1: Localisation accuracy experiments setup

Likewise, for differential GPS, the laptop served as the base station and the wearable served as the mobile unit, in a manner similar to the one described by Valejo *et al.* [233]. The initial GPS reading of the base station was considered as the positional reference from which subsequent readings were subtracted to deduce the error due to drift. The resulting error was transmitted to the wearable over the wireless link and subtracted from the mobile unit's detected position for each reading (fig. 6.1.b). The experimental procedure was otherwise the same as with the

[3] All the values presented in the accuracy experiments are in metres, resulting from the transformation of latitude and longitude to local coordinates.

standard GPS unit.

The significance of each test was deduced by hypothesising that each test has a mean equal to the 'ideal' case, namely $\mu_0 = 0$ m for the stationary tests and $\mu_0 = 30$ m for the distance tests. The deduced significance level shows the probability that the samples obtained are consistent with these assumptions. Nonetheless, this analysis does not demonstrate the performance of the system with time. This is clearly apparent from the graphs, which depict the drift of each reading over time.

6.3.2 Results and discussion of localisation measurements

GPS stationary test

The first tests aimed to determine the positional stability of the system using the handheld GPS unit. The system was set to detect its position every second over about ten minutes. When performing tasks such as mapping, it is normal to integrate GPS readings over significant periods of time so that random fluctuations (due to changes in path length etc.) are reduced. This is obviously not feasible in the context of this work as positional fluctuations cannot be distinguished from genuine movements of the user; and fluctuations in position while the user is stationary certainly disrupt the feeling of presence (see Chapter 1). The GPS drifts for X and Y are shown in figure 6.2.

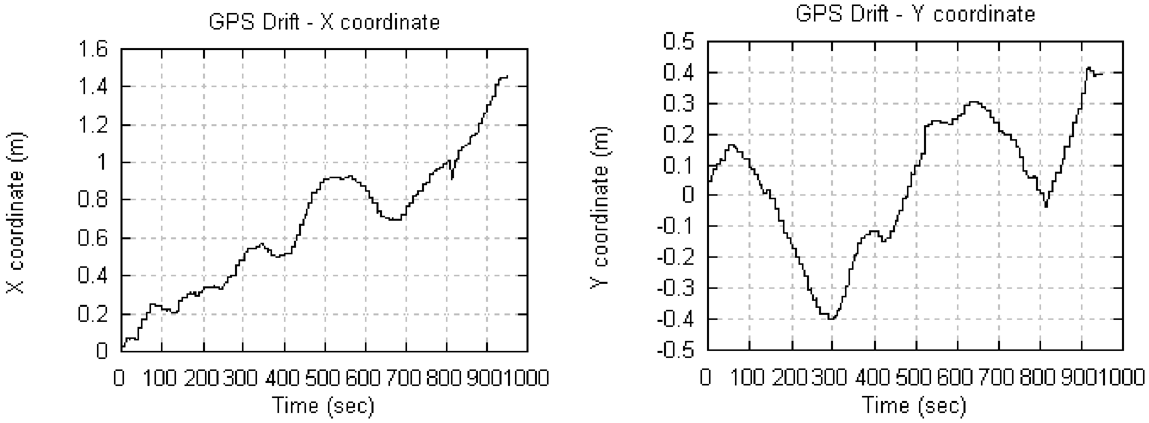


Figure 6.2: Standard GPS X and Y coordinate drift while stationary

The measured fluctuations in the X coordinate can be as severe as 1.4 m, whereas Y fluctuates between ± 0.4 metres, This introduces an observable, unrealistic movement of the virtual objects. The variation of the calculated mean from the ideal value of $\mu_0 = 0$ for both coordi-

	GPS X	GPS Y
Mean ($\pm S.D$)	0.55±0.27	0.01±0.21
$t(\mu_0 = 0)$	56	1.61
$P(X \neq X_{ideal})$	0.99	0.90

Table 6.1: GPS Drift — analysis profile

nates, is significant to a level of 99.9% for X and to 90% for Y, as shown in Table 6.1. It is safe to assume that fluctuations such as those measured here, will disrupt the feeling of presence [115, 183, 202] (see Chapter 1), especially if there are virtual objects near the user. Figure 6.3 demonstrates the effect of these fluctuations in position in the tour guide application. Although the user remained stationary the figure shows the path traversed in the virtual world due to the GPS drift.

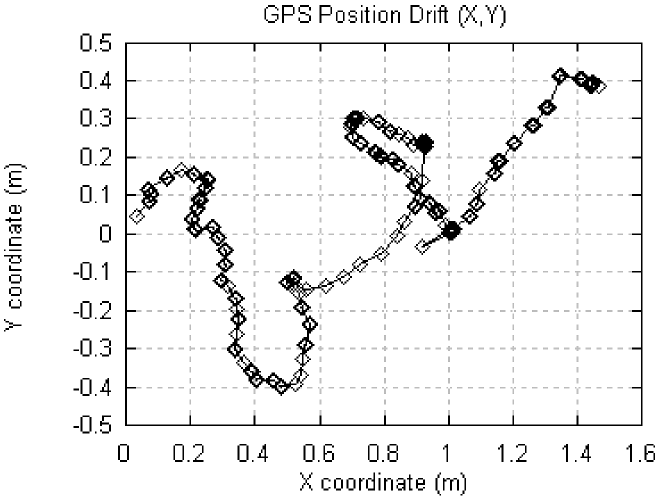


Figure 6.3: Standard GPS drift while stationary — combined positioning

On the other hand, the level of accuracy observed is adequate for demonstration systems where rough 'registering' is required. The t-tests with a hypothesised mean of zero ($\mu_0 = 0$) for both the X and Y coordinates show that the actual values measured probably differ from zero, yet the difference seems to be within 1.5 m. This leads to the likelihood that, although the stationary tests may not average to zero, position will be within 2 m of its correct value.

Generally, the GPS drift seems to remain around this level while an adequate number of satellites (> 4) are visible. For applications that require coarse positioning, like the virtual information tags on buildings used in the Touring Machine [71], this level of accuracy may be adequate. For wearable AR that employs 3D architectural reconstructions the values are large and, although comparable to other wearable AR systems, they are far greater than those required in AR. This demonstrates that wearable computing paradigms based around GPS systems cannot provide the desired positional accuracy for realising effective and precise AR systems for 3D architectural reconstruction.

GPS distance test

An equally-important requirement for localisation is to be able to detect positional displacement due to a user's movement. As the user walks in a particular direction, the system should detect the movement properly, and the resulting travel in the virtual and real worlds should be identical. The aim of this test is to determine the localisation sub-system's performance in this respect.

The author, wearing the wearable computer, walked a measured distance of 30 m on flat ground in various fixed directions. The system logged the GPS readings during the travel and in particular the final measurements, from which the distance travelled can be calculated using Pythagoras' theorem.

The experiment was performed thirty times, walking in various different directions, and the resulting distances were computed and the results plotted (fig. 6.4). They have a statistical profile shown in Table 6.2.

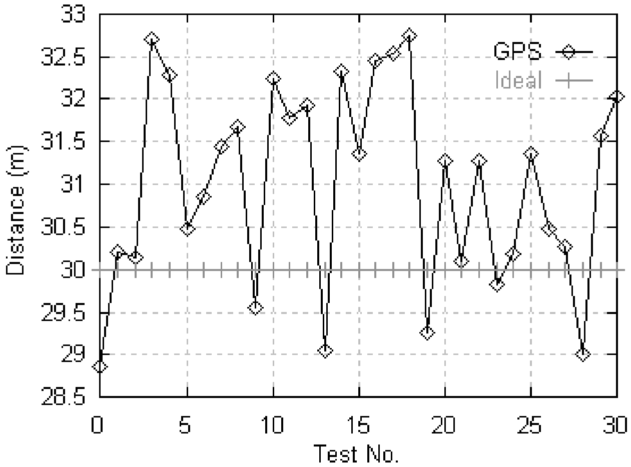


Figure 6.4: Graph of calculated distance from origin

Just as in the case of the stationary experiments, where the position drifts with time, the resulting distance is within ±2.5 m of the real value and the calculated mean is different from the hypothesised value ($\mu_0 = 30$ metres) with a probability of 99.9%. The performance of the

	GPS
Mean ($\pm S.D$)	31±1.18
$t(\mu_0 = 0)$	4.65
$P(D \neq D_{ideal})$	0.99

Table 6.2: GPS distance test — analysis profile

system is such that the distance at any measurement can be off-the-mark by more than 2.5 m.

One of the problems that has been observed during testing is a ‘hysteresis’ effect. Let us assume that the user initialises the application at a predefined position which is considered as the global scene origin. If the user walks for a few metres and returns to the origin, the virtual and real world coordinate system positions will not be identical. Just as in the stationary tests, the values obtained show that GPS is adequate for coarse positioning (within 2 m), yet it is not accurate enough for fine positioning.

DGPS stationary test

The DGPS mechanism that was used in these experiments was described in Section 6.3 and the experimental procedure was the same as for standard GPS. The results obtained from the stationary tests were plotted (fig. 6.5) and the statistical profile is shown in Table 6.3.

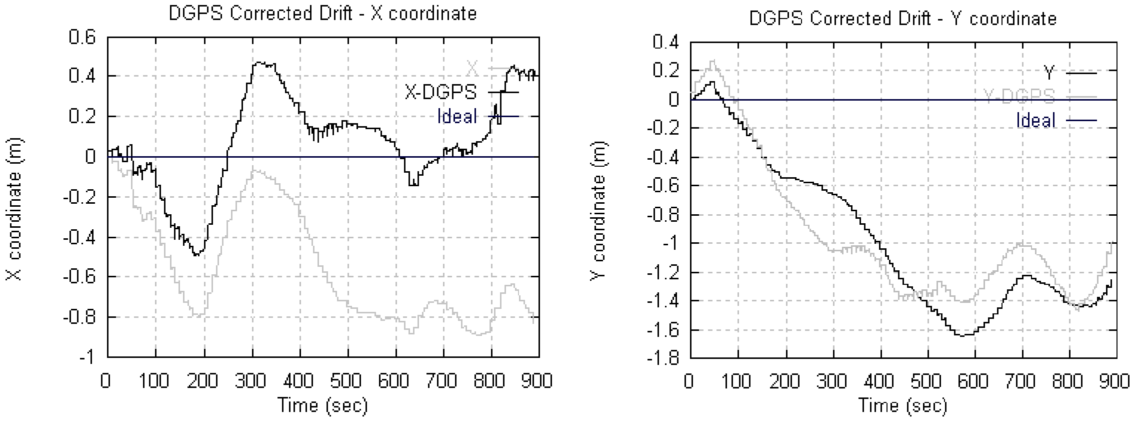


Figure 6.5: Differential GPS X and Y drift while stationary

The behaviour is similar to that obtained using standard GPS. As far as the X coordinate is concerned, there is some improvement compared to the standard GPS values: the average value for X_{DGPS} is closer to zero than for X_{GPS}. The Y coordinate values are more or less similar to the GPS values. The real-time performance of the system has improved to some extent for

	GPS X	**GPS Y**	**DGPS X**	**DGPS Y**
Mean $(\pm S.D)$	-0.55 ± 0.28	-0.96 ± 0.53	0.07 ± 0.24	-0.94 ± 0.47
$t(\mu_0 = 0)$	-58.66	-54.1	8.71	-59.73
$P(X, Y \neq X, Y_{ideal})$	0.99	0.99	0.99	0.99

Table 6.3: GPS drift — analysis profile

the X coordinate, with fluctuations for X_{DGPS} between ± 0.4 m while X_{GPS} fluctuates between 0-0.8 m. No improvement is observed for the Y coordinate. These experiments were repeated to validate these results, yielding similar drift and average values with no improvements observed by using DGPS.

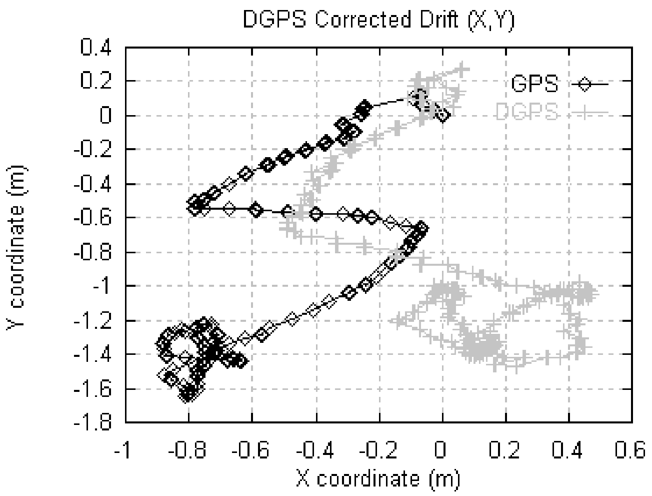


Figure 6.6: Differential GPS drift while stationary — combined positioning

Figure 6.6 shows the drift due to the combination of X and Y with time. It is apparent that this is similar to the uncorrected values obtained from GPS. Although the range of the drift is less, mainly due to the adjustment observed to the X coordinate, the use of DGPS does not improve significantly the real-time stability of the system. The drift observed results in an observable, uncontrollable movement in the virtual world, just as in the case of standard GPS. Moreover, DGPS tests showed a significant difference between the hypothesised value of μ_0=0 m and the measured one. This shows, as in the case of standard GPS, that it is probable that the 'true' mean will not eventually average to zero. It is apparent that GPS and DGPS are accurate enough for coarse positioning only.

DGPS distance test

The last of these tests deals with testing the accuracy of DGPS in detecting displacement due to movement. The same experiment as in the case of standard GPS was performed. The corrected and uncorrected values for X and Y are plotted (fig. 6.7) and the statistical profile is shown in Table 6.4.

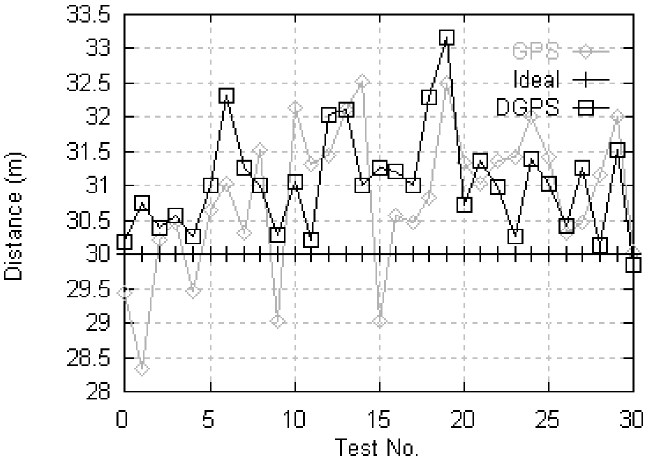


Figure 6.7: Graph of calculated distance from origin per experiment

	GPS	**DGPS**
Mean $(\pm S.D)$	30.84±1.04	31.04±0.75
$t(\mu_0 = 30)$	4.41	7.59
$P(D \neq D_{ideal})$	0.99	0.99

Table 6.4: DGPS distance test — analysis profile

The results obtained exhibit similar behaviour for both GPS and DGPS. The mean and the variance of both tests is almost the same. Both groups of results show a high significance level (99%) of difference between the measured mean and the hypothesised ideal mean of $\mu_0 = 30$ m. Nonetheless, the real-time performance for DGPS appears to be, to some degree, better than for standard GPS. By observing figure 6.7, the reader can see that more of the distance values obtained from DGPS are closer to the ideal value of 30 m.

6.3.3 Summary of GPS/DGPS experiments

The results obtained from this investigation lead to a series of several conclusions. The first is that there is a small advantage in using the homegrown DGPS solution over standard GPS, as far as fine, real-time accuracy is concerned. Some minor improvement was observed in some cases from using DGPS, yet the desired level of stability was not obtained.

The advantages in real-time position stability using DGPS over GPS are small. Although this is a surprising result, bearing in mind findings from other researchers, the reader should note that the solution is not a fully-featured DGPS system but rather a low-cost one similar to that described by Valejo *et al.* [233]. Discrepancies between the two handheld units and synchronisation problems may have led to the decreased accuracy of this homegrown solution. However, the Garmin unit has DGPS functionality, accepting corrections in RTCM 104 v.2.0 format. To accommodate that, minor modifications in the application would be required to accept NMEA input instead of the GARMIN protocol currently used. In the future, these may be incorporated into the application in order to assess modifications of the GPS sub-system.

The real-time performance of the positioning subsystem may be adequate for coarse positioning but additional, or alternative, positioning modalities would have to be used to achieve accurate registration. A drift of more than 0.2 m is too high for users to roam in the virtual space, if there are visible virtual elements surrounding the user or his or her path. Likewise, virtual components will not be able to be abutted to real-world objects. Furthermore, comparing these figures with those required for AR without considering the performance implications of wearable computers, they are high enough to induce motion sickness [74, 183, 184] and most probably disrupt any feeling of immersion [183, 202].

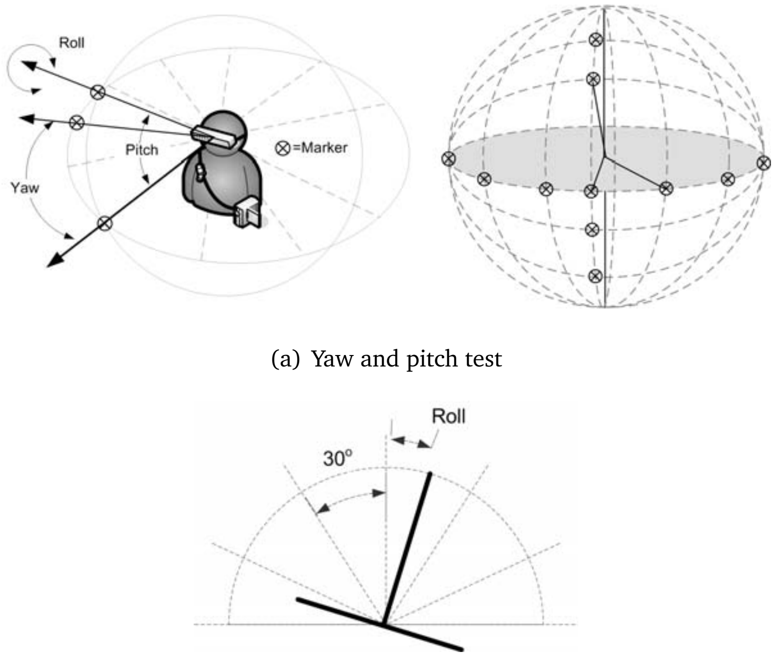
6.4 Head orientation sub-system accuracy test

6.4.1 Experimental setup

To test the orientation subsystem's accuracy two experimental virtual environments were produced. A wire-frame sphere, with a radius equal to the claimed focal length of the Virtual IO HMD (11 feet = 3.35 metres) and with 12 facets, was used to divide the virtual space into slices of 30° horizontally and vertically (fig. 6.8). Each intersection of the lines denotes a marker's position. A small crosshair, similar to the ones used in first-person-shooter games marking the point of view of the user,[4] was used to target these crossings.

Roll was similarly assessed, using a virtual semicircle with radial markers every $30°$ and origin at the centre of the focal plane, positioned in virtual space at the HMD's focal distance. An inverted T-shaped target, marking the vertical[5] and horizontal axes was used to align the HMD to the target's radial markers.

Twenty users were asked to target the points on the virtual sphere and the radial markers on the virtual hemisphere, using their dominant eye[6] to avoid binocular rivalry effects, and press a key which caused the yaw, pitch and roll to be logged to a file. The analysis of these yaw, pitch and roll measurements yields the overall accuracy of the system and the HMD 'shake' with time, similar to the time drift of GPS.



(a) Yaw and pitch test

(b) Roll test

Figure 6.8: Orientation accuracy experiments setup

6.4.2 Results and discussion of orientation measurements

Accuracy test

As outlined above, twenty users were requested to target the crossings of the wire-frame sphere and to log the HMD tracker results by pressing a key. Each user could spend as much time

[4] (Vx, Vy, Vz) in the `gluLookAt()` function.

[5] (Ux, Uy, Uz) in the `gluLookAt()` function.

[6] Determined with the method described in: `http://www.microvision.com/pdfs/nomad_domeye.pdf`

as required to target. The measurements were analysed using a t-test, showing high statistical significance levels. The results of the analysis are shown in Table 6.5 and Table 6.6.

The tracker performed fairly accurately: the errors can be attributed to the targeting abilities of each user as well as loose strapping of the HMD display on the wearer's head. The analysis shows a good accuracy for all yaw, pitch and roll angles tested, with a probability of 99.9% that the measured angle is the same as the ideal/expected ones. The accuracy of the head tracker appears to be sufficient, even for more demanding tasks than the tour guide application. The standard deviation of the values obtained is within acceptable limits (approximately $\pm 0.6°$) and in conjunction with the overall accuracy makes the Virtual IO head-tracker a valuable orientation mechanism.

Yaw Angle	$180°$	$210°$	$240°$	$270°$	$300°$	$330°$
Mean ($\pm S.D$)	179.98	210.05	240.07	269.85	299.96	330.12
S.Deviation	±0.52	±0.51	±0.54	±0.36	±0.44	±0.38
t value	−0.21	0.47	0.61	−1.89	−0.43	1.45
$P(\overline{X} = X_{ideal})$	0.99	0.99	0.99	0.99	0.99	0.99
Yaw Angle	$0°$	$30°$	$60°$	$90°$	$120°$	$150°$
Mean ($\pm S.D$)	−0.08	29.95	59.98	89.95	119.99	150.0
S.Deviation	±0.54	±0.48	±0.54	±0.45	±0.53	±0.35
t value	−0.63	−0.46	−0.13	−0.52	−0.11	−0.06
$P(\overline{X} = X_{ideal})$	0.99	0.99	0.99	0.99	0.99	0.99

Table 6.5: Yaw – analysis profile

	Roll			**Pitch**		
Angle	$-30°$	$0°$	$30°$	$-30°$	$0°$	$30°$
Mean ($\pm S.D$)	−30.06	0.03	30.06	−30.10	0.01	30.06
S. Deviation	±0.60	±0.41	±0.63	±0.58	±0.16	±0.49
t value	−0.53	0.28	0.45	−0.76	0.15	0.53
$P(\overline{X} = X_{ideal})$	0.99	0.99	0.99	0.99	0.99	0.9

Table 6.6: Roll and Pitch – analysis profile

Stability test

However, the tracker does generate some 'jitter' as the user's head remains in one orientation, resulting in a noticeable shake of the virtual world. As mentioned in Chapter 5, the source of the problem was the variation in yaw, remedied to some extent by averaging the last three values of yaw that the application receives from the tracker.

This test aims to examine the stability of the head tracker, with filtering enabled, while the HMD remains stationary. The HMD was placed on a dummy's head looking towards the 'origin' i.e., where (yaw, pitch, roll)$=(0.0°,0.0°,0.0°)$. Three thousand subsequent tracker readings for yaw, pitch and roll were logged in real time and plotted (figures 6.9, 6.10, 6.11). The analysis profile is shown in Table 6.7. From the graphs it is apparent that the HMD tracker is stable within acceptable limits. Yaw fluctuates between $\pm 0.2°$ with occasional peaks at $\pm 0.4°$. Pitch fluctuates between $\pm 0.1°$, with no sudden peaks. Roll, on the other hand, seems to fluctuate between $\pm 0.1°$ with occasional sudden peaks at $\pm 0.8°$. The analysis shows satisfactory performance in terms of the mean and the standard deviation of these values. The t-test analysis shows that the hypothesis that the true mean of the values obtained is correct has a probability of 99.9 %.

	Yaw	**Pitch**	**Roll**
Mean $(\pm S.D)$	$0.0° \pm 0.13°$	$0.0 \pm 0.06°$	$0.0 \pm 0.14°$
$t(\mu_0 = 0/0/0)$	-0.90	0.0	-0.1
$P(\theta = \theta_{ideal})$	0.99	0.99	0.99

Table 6.7: HMD shake – analysis profile

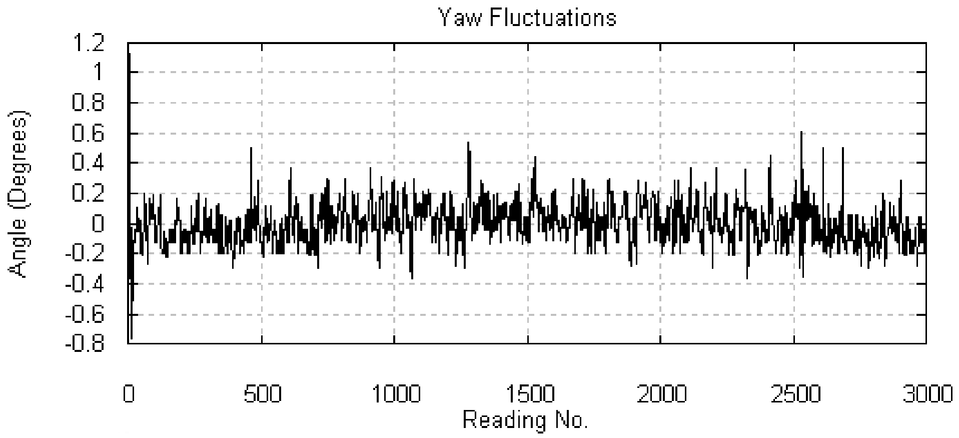


Figure 6.9: Yaw fluctuations

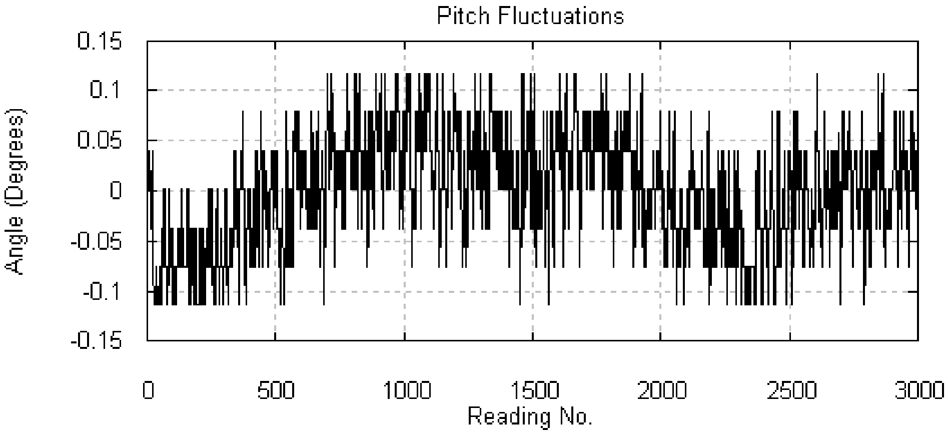


Figure 6.10: Pitch fluctuations

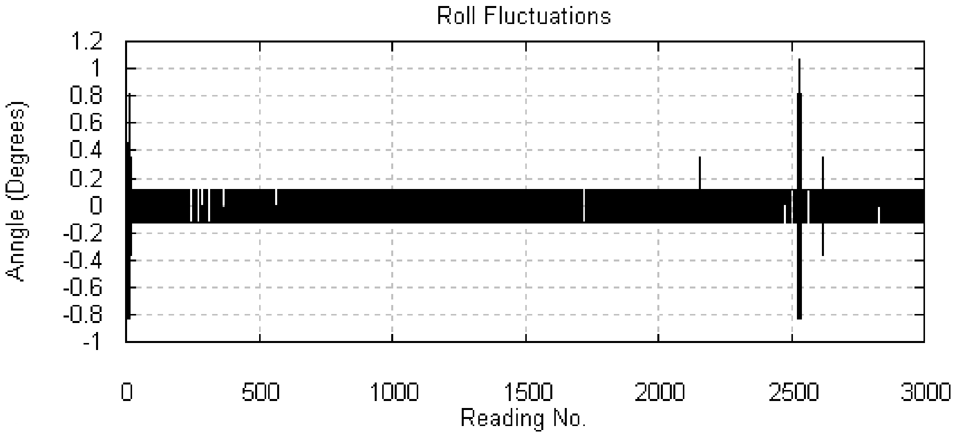


Figure 6.11: Roll fluctuations

Nonetheless, it was observed in informal tests that under normal conditions the HMD introduces further ‘shake’, believed to be due to its sensitivity, when users walk or rotate their heads. Since this is difficult to measure quantitatively, it is one of the aspects of the user satisfaction survey, described in Section 6.5

6.4.3 Summary of the orientation experiments

Because the Gosbecks Temple model is fairly big (in physical dimensions) small discrepancies may not be easily detected, at least as far as orientation is concerned if the virtual objects are in distance. Although absolute orientation errors seem to be fairly small from the above measurements, the problem is the HMD tracker’s sensitivity, which results in observable ‘shake’ of the virtual world when the user is walking.

Furthermore, it should be noted that human perception of orientation is a combination of head rotations and the direction of gaze [112, 148]. The HMD detects only head movement, so any 'shaking' effects are likely to be rather subjective and may or may not be distracting. Performing experiments for this explicitly is rather difficult; nonetheless, by questioning users through a user assessment survey like the one presented in the next section on how they feel the HMD performs, useful information can be obtained.

The following sections describe an *in situ* user assessment survey, conducted in order to investigate the usability of the Romulus wearable and the AR tour guide application described in Chapter 5. The methodology, results and discussion sections are presented, in accordance with recommendations from the APA Publication Manual [11].

6.5 User assessment survey

The purpose of the survey was to assess various aspects of the Romulus system and tour guide application, derive information on the user's perception of the system's response and accuracy in determining orientation and location, their opinion on the interface and ergonomic issues of the wearable and HMD. Their sense of presence (see Chapter 1) was also to be investigated.

Designing of the questionnaire content was carried out in a similar fashion as the one introduced by Ames *et al.* [8]. The main challenging aspects of AR systems were determined through the literature reviewed and presented in the first three chapters. A similar approach was followed for wearable AR systems. These aspects and issues were recorded and were subsequently used as the major categories the questionnaire was to examine. Examination of other research efforts in AR/VR based on questionnaires, such as the ones from Ames *et al.* [8], Billinghurst *et al.* [34], Kalawsky *et al.* [114, 115] and Siegel and Bauer [205], also pinpointed aspects that need to be evaluated and identified tried questionnaire designs.

6.5.1 Pilot survey

A pilot questionnaire, presented in Appendix I, was designed and used in a pilot survey with 10 participants. The participants were asked to wear Romulus, roam on the football pitches while immersed in the Gosbecks temple application and subsequently answer the questionnaire[7].

The pilot questionnaire consisted of 24 questions, of which two were dichotomous (YES/NO response format) and the rest used a Likert scale [70, 200] (Strongly agree – Strongly disagree)

[7] The procedure of the experiment was similar to the one described further in the chapter in Section 6.6.

for the responses. All questions were positively biased. The experiment's details and the user's personal details were the first elements recorded, followed by sections on wearable ergonomics, visual output realism and quality and general remarks.

A series of shortcomings were identified in this pilot survey. Some of the questions included terms, such as 'rendering', 'field-of-view' and 'resolution', which may be difficult for inexperienced users to understand. Ideally, simple wording and no form of leading questions [42, 70, 200] are to be used, or when attitude scales are used for the answering scheme, alternating biasing should be employed [56, 76, 181].

The use of the Likert scale resulted in neutral, or mid-point, response on some questions, not allowing any meaningful conclusions to be drawn. The 'neither agree nor disagree' response may indicate that the users were apathetic and indifferent [64], or that they did not understand the question due to difficult wording. However, the inclusion of a mid-point is a continuing debate among researchers of survey practices and has supporters and detractors, the latter claiming that the results obtained with mid-pointed scales are uninformative [152]. Supporters, on the other hand, argue that the inclusion of the mid-point avoids artificially creating a directional opinion [55, 63, 176, 215]. Converse and Presser [55] add that a middle option makes a difference to participants that do not feel strongly about the other statements, whereas participants with a strong opinion are affected much less. Furthermore, the use of the Likert scale may result in response acquiescence [76, 181], i.e. the tendency for passive assent or agreement from the users, when all answering scales have the same direction (as in the pilot questionnaire). It is advisable to use a 'balanced' response scale, i.e. items directly worded but scales worded in ascending level of rating, acceptance or agreement in both left and right directions [27].

Last but not least, after completion of the pilot survey the content of the questions was re-evaluated to include a broader set of wearable AR issues as the original set did not investigate aspects such as scale and depth perception, field of view and the participants' perceived sense of presence.

6.5.2 Finalised questionnaire

Reassessing the questionnaire content through the aforementioned literature led to a more complete and focused set of aspects that needed to be examined. A diagram of the resulting content associations of the aspects to be examined is shown in Figure 6.12. The questionnaire was re-designed and reworded with the help of two experts from the Departments of Psychology and

Linguistics of the University of Essex (questionnaire provided in Appendix K). Content validity was assessed from two academics and two doctoral candidates with experience in AR/VR research and wearable computing. The questionnaire was pilot-tested by 10 participants and no problems were identified.

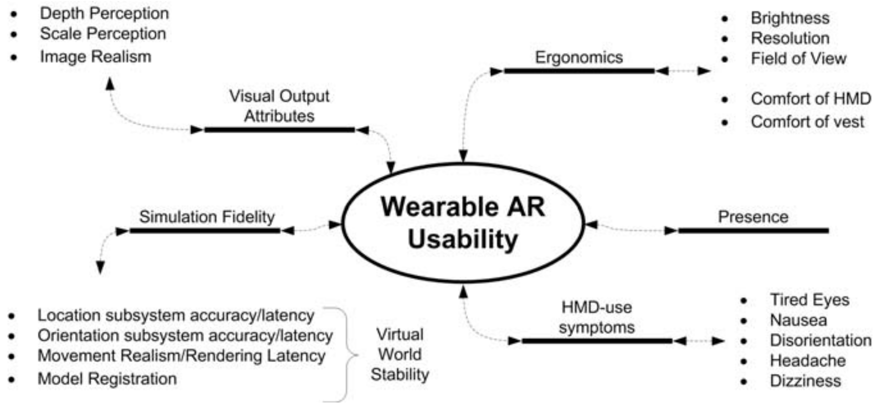


Figure 6.12: Questionnaire Content Design

The new questionnaire contains a combination of open-ended and fixed-choice questions. Four exploratory open-ended questions attempt to collect participants' general comments on the wearable computer vest and the HMD. The reason for including them was that users could describe problems, deficiencies or other remarks in their own words and without using technical terms. This is done prior to reading the rest of the questionnaire so that participants are not led from the fixed-choice questions.

A set of 21 further questions follows the open-ended section. Different Likert-type scales are used for the answers, with an alternating direction to avoid response acquiescence [27]. The wording of most answering scales was modified to reflect the content of the question. The mid-point answer is retained for this stage of the assessment, and depending on the results and reliability of the questionnaire it may be modified in future assessments (see Section 6.8). Most midpoints do not infer a 'neutral' answer but an average rating of quality, ease of use etc.

Any technical terminology was replaced by simpler statements in an effort to make the questionnaire addressable to non-technical and inexperienced participants. The first five questions investigate the most common symptoms of HMD use, as described in [8]. The rest of the questions address the aspects shown in Figure 6.12.

6.6 Method

6.6.1 Design

Assessing the usability of the Romulus wearable computer and the Gosbecks tour guide application was based on a field study. A field study in the context of this thesis is a method of collecting user feedback using a questionnaire, by going to a site of importance to the function of the wearable AR system and resembling normal operating conditions.

The survey was conducted on the University of Essex Campus and involved recruited participants from the university's population, i.e. staff and students. The participants used the wearable computer for small periods of time (≈ 15 min), under clear[8] daylight [108, 230] conditions (average measured luminous intensity of grass in experiment area $\approx$ 4000 cd/m^2), roaming in the virtual Gosbecks temple, superimposed on the sports pitches. Upon completion of the experiment each user was given the questionnaire and was asked to fill it in.

6.6.2 Participants

The application–wearable pair was assessed in terms of its usability by a convenience sample of students and academics, recruited on the university campus. Forty participants, 19 female (47.5%) and 21 male (52.5%), of ages ranging from 19 to 54 (Table 6.8) were asked to complete the post-experiment questionnaire. A brief description of the project (provided in Appendix J) was given to each of them in advance of the experiment. The only prerequisite was that all participants had normal (natural or corrected) eyesight, as this can affect their assessment of the visual output.

With respect to their computer experience, 20% regarded themselves as experts, 20% as advanced users, 25% as intermediate and 30% as novice users. 5% claimed that they had no experience. 82.5% claimed that this was the first time they used a HMD, and 90% reported that this was the first time they had used a wearable computer.

6.6.3 Materials and apparatus set-up

Location

The survey took place on the sports pitches of the University of Essex, which resemble the Gosbecks site in being flat and unobstructed. Assessment at the actual site was not easy because

[8] Fine to partly cloudy with distinct solar corona.

	Frequency	%	Valid %	Cumulative %
<= 19	1	2.5	2.5	2.5
20–24	11	27.5	27.5	30
25–29	14	35	35	65
30–39	7	17.5	17.5	82.5
40–49	3	7.5	7.5	90
50+	4	10	10	100
Total	40	100	100	

Table 6.8: Age groups of participants

it is a fair distance from the University and arrangements for transport were difficult during term time and would be inconvenient for participants. The fact that in the real field there are no remains of the temple led to the assumption that it would be adequate to test the system at a similar location. Furthermore, informal tests had shown that the number of GPS satellites detected at both locations is similar (>4 most of the time). The same region of the sports area was used as the starting point for all participants. Initial orientation was always the same, facing east.

System set-up

The experiment was based on the use of the Romulus wearable computer described in Chapter 4. This was running the tour application described in Chapter 5. The application was initialised during boot time so the users were not required to input any commands. During the experiment, no keyboard or mouse was required either. The HMD's visor was replaced by one with neutral density filters (Visor 2 — Appendix F) to reduce the amount of light from the real world reaching the eyes.

6.6.4 Procedure

Upon obtaining informed consent of participation, users were asked to wear the system in the testing area, where a small familiarisation with the main components was carried out without actually powering–on the wearable. Users were shown the wearable's main components and any questions on system functionality were discussed and answered. Subsequently, each user roamed around the model for 15 minutes, from cold start, i.e. no user saw the normal operation of the system prior to the test. Users were asked to walk in any way they saw fit, in order to put the effort to identify visual landmarks in the model themselves. Upon completion of the test each user was given a questionnaire, described in the next section, and asked to fill it in on-site.

6.7 Results of user assessment survey

6.7.1 Initial analysis

Statistical analysis of the data was carried out using Statistical Package for the Social Sciences (SPSS) version 13.[9] Due to the ordinal level of the data, results are presented using mode and variation ratio and analysed using non-parametric tests [56, 110, 206] such as Spearman's ρ correlation coefficient and Kendall's tau-c test of association. Internal consistency and (scale) reliability testing was performed using Cronbach's α [59]. The results and their implications are discussed in Section 6.8.

The responses of participants were coded as 5–1, positive to negative in respect to the overall usability; Question 14 (Q.14) has an extra scale level of 'Not applicable' implying a 'No' response to Q.13. The questions were subsequently summed to a total score reflecting the overall usability of the system. A maximum of 96 and a minimum of 19 represent excellent and very poor usability respectively. The total scores obtained from each participant are presented in fig. 6.13 and their descriptive statistics in Table 6.9. The average score of the 40 participants is 76.2 ± 6.8.

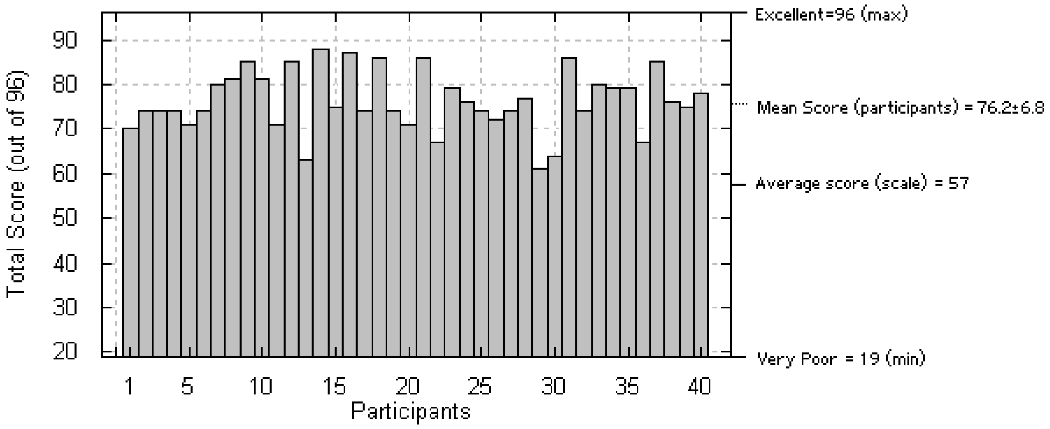


Figure 6.13: Total scores obtained per participant

	N	Min	Max	Mean	S.D.	Mode	Median
Total	40	61	88	76.2	6.8	74	75
Valid N	40						

Table 6.9: Total score statistics

[9] http://www.spss.com/

The reliability of the scale was determined to be satisfactorily reliable (Cronbach's $\alpha = 0.786$). Table 6.10 presents the reliability analysis statistics. It is noteworthy that deletion of Q.14 (resolution) could improve the overall reliability. Q.14 assesses the resolution of the HMD in conjunction with Q.13 (Yes/No) in an attempt to simplify wording. However, in the analysis only Q.14 is used. Arguably, the fact that the reliability could improve by changing or omitting it indicates that different wording and coding scheme may be more appropriate. Deletion of any of the other questions does not appear to increase significantly the overall reliability of the scale.

	Scale Mean if item deleted	**Scale Variance if Item deleted**	**Corrected Item-Total Correlation**	**Squared Multiple Correlation**	**Cronbach's α If Item Deleted**
Q.5	66.70	44.882	.096	.523	.789
Q.7	66.28	44.410	.318	.499	.782
Q.8	66.25	45.474	.068	.365	.788
Q.9	66.78	43.820	.171	.732	.787
Q.10	67.20	41.241	.426	.783	.772
Q.11	67.40	41.272	.519	.669	.768
Q.12	67.45	40.305	.489	.652	.767
Q.14	66.35	41.669	.079	.347	.825
Q.15	68.50	44.000	.109	.447	.792
Q.16	68.33	39.815	.480	.760	.767
Q.17	68.40	39.990	.394	.480	.774
Q.18	67.78	39.051	.549	.686	.762
Q.19	67.35	39.362	.663	.732	.757
Q.20	67.38	41.881	.450	.619	.772
Q.21	67.35	40.490	.472	.608	.769
Q.22	67.98	44.179	.138	.558	.788
Q.23	68.98	42.487	.341	.516	.777
Q.24	67.43	38.251	.630	.745	.756
Q.25	67.30	37.549	.786	.871	.746

Table 6.10: Item-total statistics

Symptoms

A series of symptoms of HMD use were assessed. 52.5% of the participants reported slightly tired eyes and the remaining no tiredness at all (Q.5). None of the participants felt nauseated (Q.6). Only 10% experienced slight headache while using the system (Q.7) and only 7.5% felt dizzy (Q.8). 10% felt moderately disoriented, 40% slightly disoriented and the remaining not disoriented at all (Q.9). The results are presented in Figure 6.14 and Table 6.11.

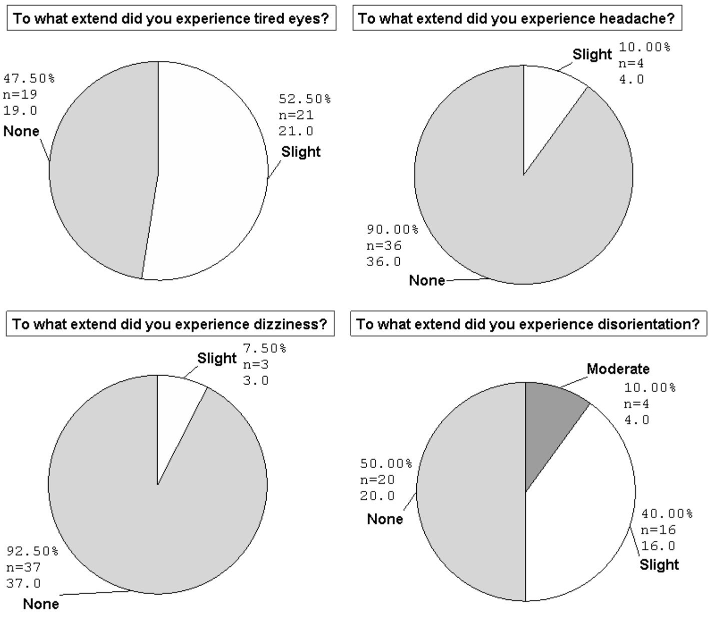


Figure 6.14: Reported symptoms of HMD use

To what extend did you experience:	N	Min	Max	Mode	Var. Ratio
Q.5 tired eyes	40	4	5	4	.475
Q.6 nausea	40	5	5	5	0
Q.7 headache	40	4	5	5	.100
Q.8 dizziness	40	4	5	5	.075
Q.9 disorientation	40	3	5	5	.500
(Scale: None=5, Slight=4, Moderate=3, Severe=4, Very Severe=1)					

Table 6.11: HMD symptom score

Ergonomics

Ergonomic issues involved the comfort of use of the HMD, the wearable computer and characteristics of the HMD (Table 6.12). 55% of the participants felt that the comfort of the wearable was good and 22.5% that it was excellent (Q.10). 20% felt that it was adequate for the task and 2.5% that it was poor. Likewise, for the HMD comfort assessment, 57.5% thought it was good, 32.5% adequate and only 10% felt that it was excellent (Q.11).

The HMD characteristics investigated were the resolution, brightness and field of view. 62.5% of the participants claimed that they could not distinguish the pixels of the screen (Q.13 and Q.14). Some 22.5% thought they were not very noticeable whereas 10% did not find them noticeable (Q.14). As far as HMD brightness is concerned, 42.5% felt that it was adequate, whereas 37.5% claimed that it was dim (Q.15). 15% felt that it was bright and 5% that it was too dim for their liking. Last but not least, 47.5% of the participants claimed that they became aware of the edges of the screen but that they were not very noticeable (Q.16). 27.5% claimed that the borders were quite noticeable and 5% that their view was noticeably restricted. Conversely, 17.5% felt that their view was not noticeably restricted and one participant (2.5%) claimed that the view was perfectly natural.

	N	Min	Max	Mode	Var. Ratio
Q.10 How do you rate the comfort of the body-mounted computer?	40	2	5	4	.450
Q.11 How do you rate the comfort of the head-mounted display?	40	3	5	4	.425
Q.14 If yes, how noticeable were they?	40	2	6	6	.375
Q.15 Would you say that the display was: (brightness level)	40	1	4	3	.575
Q.16 Did you notice ... restricted by the edges of the display?	40	1	5	3	.525
Q.10–Q.11 Scale: Excellent=5, Good=4, Adequate=3, Poor=4, Very Poor=1)					
Q.13 Scale: Yes=1, No=2					
Q.14 Scale: 5–1 positive to negative rating (see Appendix K) plus 6=Not applicable					
Q.15–Q16 Scale: 5–1 positive to negative rating (see Appendix K)					

Table 6.12: Ergonomics questions statistics

Visual output

A series of questions focused on the quality of the visual output, mainly of the virtual world, assessing ease of orientation while in the virtual world, ease of real–virtual world composition, virtual world realism and sense of depth and scale (Table 6.13).

55% of the participants indicated that it was easy to find their way in the virtual world, 25% felt that it was average whereas 12.5% that it was very easy (Q.12). 7.5% felt that it was difficult. Interestingly, 50% of the participants claimed that it was difficult to combine the real world with the virtual (Q.17). Conversely, 27.5% felt that it was easy, 17.5% that it was average and equal percentages (2.5%) were obtained for the two extremes of the scale (very difficult, very easy) . 45% of the participants felt that the realism of the virtual world was good and 40% that it was adequate for the task (Q.18). The remaining were distributed evenly (5%), over the remaining categories (excellent, poor, very poor). 62.5% of the participants reported that their sense of depth in the virtual world was good and 12.5% that it was excellent (Q.19). 20% claimed that their depth sense was adequate whereas two users (2.5%) indicated that it was poor. 67.5% of the participants claimed that their sense of scale of the virtual model was good and 7.5% that it was excellent (Q.20). The participants' percentage that felt that their scale sense was adequate was 22.5% and one user (2.5%) that it was poor.

	N	Min	Max	Mode	Var. Ratio
Q.12 How easy was it to find your way around in the virtual world?	40	2	5	4	.450
Q.17 How easy to combine ... computer-generated scenes together?	40	1	5	2	.500
Q.18 How do you rate the realism of the graphical output	40	1	5	4	.550
Q.19 How do you rate your sense of depth ... while using the system?	40	2	5	4	.375
Q.20 How do you rate your sense of scale ... while using the system?	40	2	5	4	.325
Q.12,Q.17 Scale: Very easy=5, Easy=4, Average=3, Difficult=2, Very difficult=1					
Q.18–Q.20 Scale: Excellent=5, Good=4, Adequate=3, Poor=2, Very Poor=1					

Table 6.13: Visual output questions statistics

Simulation fidelity

Questions on simulation fidelity mainly addressed the stability of the virtual world when users remained stationary, walked or moved their head while stationary (Table 6.14). In order for the virtual world to remain stable the orientation and localisation sub-systems ought to function accurately and with small delays.

60% of the participants felt that the world remained quite stable when they remained still (Q.21) whereas 17.5% felt that there was a little instability. 15% felt that it was completely stable whereas 7.5% that it was fairly unstable. 55% of the participants observed a little instability of the virtual world when moving their heads while 32.5% felt that it was quite stable (Q.22). 12.5% reported that it was fairly unstable. Interestingly, 77.5% of the users reported that the virtual world was fairly unstable when they walked, yet 10% felt there was little instability (Q.23). 7.5% felt that it was quite stable, whereas 5% that it was completely unstable.

	N	Min	Max	Mode	Var. Ratio
Q.21 How stable was the virtual world when you stayed still?	40	2	5	4	.400
Q.22 How stable was the virtual world when you moved your head?	40	2	4	4	.450
Q.23 How stable was the virtual world when you walked around?	40	1	4	2	.225
Scale: Completely stable=5, Quite stable=4, A little instability=3, Fairly unstable=2, Completely unstable=1					

Table 6.14: Simulation fidelity questions statistics

Presence and overall opinion

Arguably, the most important questions in the questionnaire, are those assessing presence (discussed further in Section 6.8) and each participant's overall impression (Table 6.15). 50% of the users reported that their sense of presence while using the system was good, 22.5% that was adequate and 17.5% that was excellent whereas 10% reported it was poor. 52.5% of the users reported a good impression of the system and 20% an excellent. 22.5% felt that the system was adequate for its purpose, whereas two users (5%) felt that it was poor. The results are summarised in Figure 6.15.

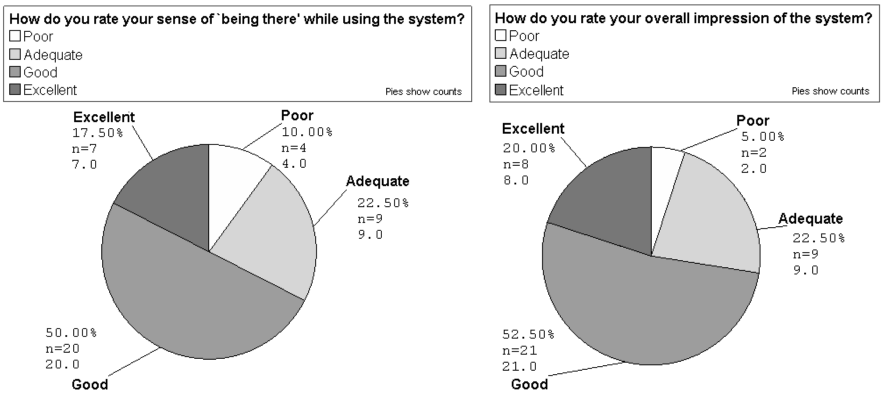


Figure 6.15: Presence and overall impression

	N	Min	Max	Mode	Var. Ratio
Q.24 How do you rate your sense of ‘being there’ while using the system?	40	2	5	4	.500
Q.25 How do you rate your overall impression of the system?	40	2	5	4	.475
Q.18–Q.20 Scale: Excellent=5, Good=4, Adequate=3, Poor=2, Very Poor=1					

Table 6.15: Presence and overall impression questions statistics

6.7.2 Inter-item correlation analysis

Because Cronbach’s α was high, it was expected that the average inter-item correlation would be high. Analysis using Spearman’s rank correlation coefficient (ρ) demonstrated sufficient correlations between most items and in particular with the overall impression. Table 6.16 presents coefficients that are significant at the $p = .01$ level only.

It is noteworthy that the participant’s overall impression (Q.25) was strongly correlated ($\rho = .740, p = .001$) with their sense of presence (Q.24). The latter is also moderately correlated with the comfort of the wearable (Q.10) ($\rho = .550, p = .001$), the scene realism (Q.18) ($\rho = .569, p = .001$), the sense of depth in the virtual world (Q.19) ($\rho = .620, p = .001$), the sense of scale (Q.20) ($\rho = .422, p = .007$) and the stability when participants remained still (Q.21) ($\rho = .418, p = .007$).

		Q.9	Q.10	Q.11	Q.12	Q.16	Q.17	Q.18	Q.19	Q.20	Q.21	Q.23	Q.24	Q.25
Q.9	ρ								.414					
	p								.008					
Q.10	ρ			.550									.405	.562
	p			.000									.010	.000
Q.11	ρ		.550											.515
	p		.000											.001
Q.12	ρ								.520					.433
	p								.001					.005
Q.16	ρ							.441						.446
	p							.004						.004
Q.17	ρ													.409
	p													.009
Q.18	ρ					.441			.418		.492		.569	.505
	p					.004			.007		.001		.000	.001
Q.19	ρ	.414			.520			.418		.481	.417		.620	.564
	p	.008			.001			.007		.002	.008		.000	.000
Q.20	ρ								.481				.422	
	p								.002				.007	
Q.21	ρ							.492	.417				.418	.542
	p							.001	.008				.007	.000
Q.23	ρ													.432
	p													.005
Q.24	ρ		.405					.569	.620	.422	.418			.740
	p		.010					.000	.000	.007	.007			.000
Q.25	ρ		.562	.515	.433	.446	.409	.505	.564		.542	.432	.740	
	p		.000	.001	.005	.004	.009	.001	.000		.000	.005	.000	
Correlation is significant at the 0.01 level (2-tailed)														

Table 6.16: Inter-item correlations (Spearman's ρ)

The overall impression (Q.25) was also moderately correlated with the comfort of the wearable (Q.10) ($\rho = .562, p = .001$) and the HMD (Q.11) ($\rho = .515, p = .001$), the ease of orientation in the virtual world (Q.12) ($\rho = .433, p = .005$), the HMD field-of-view (Q.16) ($\rho = .446, p = .004$), the realism of the virtual scene (Q.18) ($\rho = .505, p = .001$), the sense of depth (Q.19) ($\rho = .564, p = .000$) and the scene stability when the user stands still (Q.21) ($\rho = .542, p = .001$) and walks (Q.23) ($\rho = .432, p = .005$).

The ease of finding the way (Q.12) was correlated moderately with the sense of depth (Q.19) ($\rho = .520, p = .001$). The latter was moderately correlated to the realism of the graphical output (Q.18) ($\rho = .418, p = .007$), the sense of scale (Q.20) ($\rho = .481, p = .002$) and the stability when participants remained still (Q.21) ($\rho = .417, p = .008$). Last but not least, the realism of the virtual world was moderately correlated with the HMD's FOV (Q.16) ($\rho = .441, p = .004$) and the stability when participants remained still (Q.21) ($\rho = .492, p = .001$)

Of particular interest was the level of correlation, if any, of the questions examining HMD-use symptoms with the overall impression and the sense of presence. Only the symptom of disorientation (Q.9) was moderately correlated ($\rho = .414, p = .008$) with the sense of depth. However, at the p=.05 level, low correlation was observed between disorientation, the ease of finding the way (Q.12) ($\rho = .359, p = .023$) and the sense of presence (Q.24) ($\rho = .381, p = .015$). Also, headache (Q.7) demonstrated low correlation with virtual world stability when staying still(Q.21) ($\rho = .357, p = .024$) and overall impression(Q.25) ($\rho = .351, p = .026$).

6.7.3 Analysis of the open-ended questions

Responses to the open-ended questions of the questionnaire were processed using content analysis by grouping the participants' responses in categories. The purpose of these exploratory questions was to derive some feedback from the users, outside the defined boundaries of the rest of the questionnaire [58, 199]. However, most participants commented on issues examined in the fixed-choice section. In addition, a high number of missing values (no response) were observed, particularly to questions 3 and 4. This is problematic for examining accurately any correlation between the open-ended and fixed-choice questions.

Q.1 Do you have any comments about the head mounted display?	
	Count
Dim	5
Colours not intense	5
Small FOV	9
Perfect FOV	1
Low resolution	1
A bit heavy	5
Quite comfortable	2
Harnessing was loose	2
Long cable	1
Good response to head movement*	2
Average response to head movement	1
Smoothing required*	1
Fairly stable	2
Missing/No comment	15

*VR and graphics programming expert's comment

Table 6.17: Open-ended questionnaire analysis — I

Q.1 asked participants to comment on the HMD (Table 6.17). Many participants identified as problems the brightness, colour intensity and the field-of-view of the unit. Five participants felt that it was a bit heavy. Noteworthy are the comments of the expert users which commented well on the system's response to head movement, however they indicated that further 'smoothing' (filtering) could improve performance even further.

Q.2 asked participants to comment on the body-mounted (wearable) computer. Most participants focused on weight and comfort, as presented in Table 6.18, whereas a few provided more detailed comments such as vest size and garment quality considerations and wiring. Some participants pinpointed that their sense of comfort was down to the distribution of weight of the vest, noting particularly the heavy pockets (battery pack). One of the experts having used in the past the Rome wearable [157] commented that Romulus is an improvement.

Q.2 Do you have any comment about the body-mounted computer?	
	Count
Light	3
Heavy	5
Weight well distributed	1
Got used to the Weight	3
Heavy at the pockets	3
Quite comfortable	5
Correct size	1
Large size	1
Improvement over Rome [157]*	1
Fine/Nice	4
Vest quality could be better (garment)	1
Could have less wiring	1
Missing/No comment	17

*VR and Graphics Programming Expert's comment

Table 6.18: Open-ended questionnaire analysis — II

Q.3 attempted to investigate any unpleasant sensations the participants might have had while using the system (Table 6.19). The question was expected to be correlated to the symptoms questions in the fixed-choice section (Q.5–Q.9) and provide comparable insight. However a limited number of participants reported any unpleasant sensations in question 3, compared to a large number of HMD-use symptoms responses in the fixed-choice section.

Finally, Q.4 prompted for further comments, if any, on the wearable system (Table 6.20). Participants commented on their overall impression of the system, indicating a positive attitude towards it, whereas many pinpointed the instability of the virtual world when they walked.

Q.3 Describe any unpleasant sensations you may have felt while using the system	
	Count
Tired eyes	1
Slight disorientation	2
Tired from HMD weight	1
None	13
Missing/No comment	23

Table 6.19: Open-ended questionnaire analysis — III

Q.4 Any other comments?	
	Count
The components seem old and outdated*	1
Would like to see similar systems to museums and achæological sites	1
Nice/Fine/Interesting	2
Very enjoyable	3
Impressive	2
Slight disorientation after use	1
Model unstable when walking	5
Made me look funny	2
Missing/No comment	26

*Graphics programming expert's comment

Table 6.20: Open-ended questionnaire analysis — IV

In order to investigate further the implications of the participants' responses to the open-ended questions these were separated in two main variables. The first variable represents the response set in the open-ended questionnaire. The responses were classified into 'negative comment', of any type and level, 'no negative comments at all' and 'missing' (denoting that the participant has not responded in any of the four questions). Another variable represented whether participants reported symptoms in the fixed-choice questions (of any type and level) or not. The significance of the association between reported overall negative comments and reported HMD symptoms was examined using Kendall's tau-c, indicating weak and not statistically significant association between the two groups (tau-c$= -.188, p = .122$) (Table 6.21).

Following a similar approach, the relationship between negative comments on the open-ended questions and the overall impression was also investigated. One variable was similar as above, holding the response set to the open-ended questions. The other held the response set of Q.25 (Overall Impression). The association between them was also weak and not statistically significant (tau-c $= .054, p = .656$) (Table 6.22).

		HMD symptom		Total
		Yes	No	
negative comment	Yes	16	7	23
	No	1	2	17
	Missing	13	1	14
Total		30	10	40

Table 6.21: Cross-tabulation — HMD symptoms * Negative Comments

		How do you rate your overall impression				Total
		Poor	Adequate	Good	Excellent	
negative comment	Yes	0	9	9	5	23
	No	0	0	1	2	3
	Missing	2	0	11	1	14
Total		2	9	21	8	40

Table 6.22: Cross-tabulation — Overall Impression * Negative Comments

6.8 Discussion of survey results

The average score of $76.2{\pm}6.8$ represents that, overall, the 40 participants had a positive attitude towards the system, rating it as ‘Good’ (using the rating scales in the questionnaire). However, a series of deficiencies of the current system are pinpointed, that, bearing in mind the aforementioned correlation analysis, affect the sense of presence, the overall opinion and the total score. These items obtained low ratings the participants, indicating that further refinements of the system can be made.

A high percentage of participants reported some symptoms after using the HMD, however most were slight (Q.4 & Q.5–Q.9). 10% of the participants felt a moderate disorientation, whereas tired eyes, dizziness and headache were reported as ‘slight’ or ‘none’. None of the participants felt nauseated. The system seems to perform adequately, without inducing any high degree symptoms, thus proving fairly usable from participants not experienced with VR/AR systems and simulators.

It is also noteworthy that there were a limited number of reported symptoms in the open-ended questions. Overall impression (Q.25) appears not to be related to any symptoms, as the examination of association between negative comments and reported symptoms showed (Kendall’s tau-c= $-.188, p = .122$). As is apparent from the fixed-choice scores, people that reported some level of the examined symptoms were not dissatisfied with the system in other measures. Furthermore, the level of those symptoms was such that it did not affect the overall impression significantly.

Most of the participants felt that the computer vest was fairly comfortable (Q.10) (above average: 77.5%). Vest comfort was moderately correlated to the overall impression (Q.25) ($\rho = .562, p = .001$). Of course, further miniaturisation and lighter power sources could improve the overall weight of the system even further. The percentage that felt it was merely adequate was quite high (20%) whereas one user felt it was poor. Considering also the comments in the open-ended questions (Q.2), weight reduction, careful component arrangement and a better quality vest — in terms of garment durability — could further improve overall comfort. Also, alternative sizes may be appropriate to accommodate different-sized users. Last but not least, one of the expert participants supported the argument that the system is an improvement over Tin Lizzy variants such as Rome [157].

A large percentage felt that the HMD was quite comfortable for the task (Q.11) (above average: 67.5%), without any participant rating it below average. HMD comfort was moderately correlated to overall impression (Q.25) ($\rho = .515, p = .001$). However, other HMD characteristics pinpointed more important deficiencies. One of the major observed problems was the brightness of the HMD (Q.1 & Q.15). A large proportion (42.5% below average) of users felt that it was not bright enough or it was merely adequate (42.5%), even with the neutral density filters installed. However brightness did not seem to be correlated to the overall impression.

Most users (47.5%) became aware of the small field-of-view (Q.1 & Q.16) and, although they claimed it did not restrict their view, the author is sceptical whether this would be the same had they used the system more than once and for longer periods as he finds the field-of-view to be rather confining. Furthermore, 27.5% of the participants did state that the small field-of-view was noticeable and a further 5% that it was restrictive. It is important to note that the HMD's field-of-view (Q.16) was moderately correlated to the overall impression (Q.25) ($\rho = .446, p = .004$). Interestingly, most participants did not became aware of the low resolution of the display (Q.13–Q.14).

The fact that the aforementioned questions, apart from brightness (Q.15), are moderately correlated to the overall opinion and the total score shows that a replacement of the HMD could potentially improve the usability of the system. A HMD with wider field-of-view and brightness adjustment (with a range adequate for outdoor use) would almost certainly improve the output quality and the overall usability score. However, at the same time, it needs to be light enough to achieve comfort rating similar to the Virtual I/O i-glasses.

Overall, participants found that it was fairly easy to find their way around the virtual world (Q.12); however 25% found it to be average and 7.5% difficult. Also, the majority of users found that their senses of depth (Q.19) and scale (Q.20) were good (75% and 75% above average respectively), but that the overall realism (Q.18) in the model was not as good as they would like (40% average and 10% below average). Ease of finding the way (Q.12) was correlated moderately with the sense of depth (Q.19) ($\rho = .520, p = .001$), which in turn was moderately correlated to the realism of the graphical output (Q.18) ($\rho = .418, p = .007$).

One could argue that the realism of the virtual objects affected the ease of navigation, making difficult for users to identify landmarks and virtual objects. It is also safe to assume that this problem was augmented by the problems relating to brightness and field-of-view. One observation the author made while administering the experiment was that some participants had problems perceiving the 3D scene for a few seconds at the beginning of the experiment, perceiving it instead as 2D, subsequently adjusting to it. In addition, most participants followed similar patterns in walking into the virtual world, either by approaching and walking around the main temple peripteral, or by walking through the temple portico. Although there has been no follow-up investigation of these observations, further assessments should consider them as this could possibly identify what visual features draw the user's attention most.

Perception of the composite world (Q.18) was moderately correlated only to overall impression (Q.25) ($\rho = .418, p = .007$). The majority of users found it difficult to combine the virtual world with the real environment (Q.17) (52.5% below average and 17.5% average). This can be attributed to a number of reasons. Primarily, the neutral density filters (see Appendix F) may have affected to some extent the view of the real world. Secondly, the realism level of the model, along with the low brightness and low intensity of the colours identified in Q.2, made discrepancies between virtual and real worlds more apparent. However, the most important problem may be the requirement to focus at varying distances, those of the displayed virtual world (image plane) and the real world, a limitation of most optical see-through HMDs [18].

Participants were asked to comment on the stability of the virtual world when they stood still, moved their heads and walked. Virtual world stability encompasses various important parameters such as accurate location and orientation measurement, registration and system lag. As participants moved about, the virtual objects ought to remain stable (successful registration), even when their movement was rapid (low latency). Movement had to be smooth and accurate in order for the virtual objects to appear 'in the right place'. From the resulting percentages

(60%) it is apparent that the system was quite stable when participants remained still (Q.21). This was moderately correlated with the sense of depth (Q.18) ($\rho = .492, p = .001$) and scale (Q.19) ($\rho = .417, p = .008$). Little instability was observed when they moved their heads (55%), most probably because of the orientation detection sub-system delay and sensitivity. More importantly though, 77.5% of the users reported that the system was quite unstable while walking. This can be attributed to the sensitivity of the HMD tracker which has been designed for indoor, non-mobile use. Walking vibrations seem enough to introduce a fair amount of instability in the system, indicating that a 'cushioning' mechanism is required. The latter can be some form of filtering, like Kalman filters [112, 133], or a different tracking mechanism. Last but not least, modern and more advanced 3DOF magnetic sensors may offer increased stability and further investigation of potential replacements of the current tracker maybe worthwhile.

Inevitably related to the visual output quality and the simulation fidelity is the sense of presence (Q.24) and user's overall impressions of the system (Q.25), as pinpointed from the inter-item correlation analysis of questions in Section 6.7.2. The overall impression (Q.25) was moderately correlated with the realism of the virtual scene (Q.18) ($\rho = .505, p = .001$) and the sense of depth (Q.19) ($\rho = .564, p = .000$), which are related to the visual output, and the scene stability when the user stands still (Q.21) ($\rho = .542, p = .001$) and walks (Q.23) ($\rho = .432, p = .005$), which are related in turn to simulation fidelity. Overall impression (Q.25) was also strongly correlated ($\rho = .740, p = .001$) with the participants' sense of presence (Q.24), which in turn is also correlated to scene realism (Q.18) ($\rho = .569, p = .001$) and the sense of depth in the virtual world (Q.19) ($\rho = .620, p = .001$). In addition the questions on presence and overall opinion are characteristic of the system's overall usability and functionality. The majority of users claimed that their sense of presence was above average (50% good and 17.5% excellent). An even higher percentage had an above-average opinion of the system (52.5% good and 20% excellent).

However the aforementioned problems of the HMD and the virtual world realism affected 10% of the users, who felt that their sense of presence was poor — and 5% felt that overall the system was poor. As described above, the field-of-view (Q.16) is moderately correlated to overall impression and visual output (Q.18: realism, Q.19: sense of depth) is moderately correlated to both overall impression and presence. Arguably, improving the output quality, in terms of HMD optical characteristics and virtual world realism, could improve even further the participants' sense of presence and overall opinion. Interestingly the moderately negative comments, reported in the open-ended questions, most of which concentrated on the HMD and the vest characteristics, do not seem to be associated with the participants' overall opinion (Kendall's tau-c $= .054, p = .656)$.

Although replacing the HMD with one featuring a wider FOV and adequate brightness for outdoor use is feasible and subject probably only to cost and availability factors, improving on the model realism is more of a challenge, bearing in mind the capabilities of the hardware platform and the scene complexity. The use of shadows[10] and textures could improve the realism of the graphical output, thus increasing ease of navigation and sense of presence. Also, optimisations such as display lists could improve performance and maintain acceptable frame rates. Last but not least, there is evidence that stereo rendering may improve the perception of the virtual world [26, 180, 239].

It is important at this stage to note the importance of the sense of presence. AR is all about the illusion of 'being in' a computer-generated world [115, 183, 202], while maintaining a view of the real world. A number of attempts have been made to come up with metrics for the sense of presence. However, as pinpointed in [113], presence is a multi-dimensional parameter, related to various sub-parameters. The investigation presented here pinpointed some of these sub-parameters: the scene realism, HMD characteristics such as brightness and FOV, virtual world registration and the sense of depth. Although the attempt to 'measure' presence described here is rather superficial — since the purpose of the research is not solely to assess incisively the factors affecting the sense of 'being there' — it is adequate and indicative of whether the system fulfils its purpose.

Practical constraints

Although the survey was based on a questionnaire that proved reliable — as Crobach's α test for scale reliability demonstrated — there are certain limitations that, once considered, can provide further information on the nature of wearable AR. One of the practical constraints involved concerns the sample of participants. Although the age range is adequate to describe the potential community of users of such a system, had it been commercialised and used in practice, most of them are below 30 years of age. Also almost all have been university students or academics. Having a convenience sample of people was essential for this survey, aiming to provide an initial insight into the usability of the system. Arguably, such samples may generate results that cannot

[10] These have to be time-of-day dependant

be easily interpreted accurately [42]. However, there is support for the opposite argument that convenience samples do lead to interpretable results and are a favourable method compared to random sampling when the population pool is large [196], as in this research.

The author advises that future surveys should rely on more representative samples, increasing the age range if possible while maintaining equal percentages among age groups. In addition, participants with varying levels of education could increase the representability of the sample. Last but not least, the major requirement for assessing this system is that users have correct or corrected eyesight[11]. In order to test this reliably, it might be wise to assess the eyesight of each participant prior to the experiment, something which is beyond the capabilities of the author's department.

It is probably also safe to assume that the examined HMD-usage symptoms may become more severe when the system is used for periods longer than 15 minutes. Bearing in mind the time constraints and battery life, further investigation was not feasible here for practical reasons. Replication of the system could allow further investigation of HMD usage symptoms, including the effects of extended immersion in the model.

6.9 Summary of the survey

Overall, the system was rated by participants as 'good' in terms of its usability. Strong points of the system are the good overall comfort of the HMD and the vest, the good sense of depth of the user, the fast response to movement and change of orientation and the lack of any severe ocular and non-ocular symptoms after the use of the system. However, some problems were identified, which, once remedied should improve overall opinion and usability. These were the limited field-of-view, the inadequate HMD brightness, the realism of the virtual world and the stability of the virtual world when users walk.

The survey presented can provide a basis for further assessment of the usability of wearable AR systems. It is easy to administer, and the results obtained can be used to refine it further. Assessing ocular and non-ocular symptoms of HMD use is an important subject on its own, therefore a separate questionnaire focusing on these, along the lines of those presented by Kennedy *et al.* [120] and Ames *et al.* [8], may be more appropriate.

A few questions may have to be added in the questionnaire presented in this research as features are added to the wearable computers or application, perhaps examining application

[11] Although, arguably, potential users of a commercial system may not.

control, stereo rendering and texture and lighting realism. Exploratory, open-ended questions should be retained in order to derive 'raw' feedback from experiment participants. Finally, construction of several of wearables (>3) can reduce overall experiment duration, while allocating longer time slots per participant, thus allowing more in-depth investigation of wearable AR.

6.10 Chapter summary

Wearable AR prototypes have to face a series of performance issues relating to rendering speed, accuracy and stability of location and orientation measurement and poor ergonomics. Many systems are difficult to use and extensive surveys are difficult to carry out. To produce a platform that is practical to use, a series of experiments were carried out in order to determine the accuracy and stability of the location and orientation subsystems, as well as to investigate the usability of the system.

The first two sets of experiments involved field-based, quantitative investigations of the location determination subsystem. The GPS and homegrown DGPS solutions were assessed both in terms of the stability of the readings and their distance resolution. Using a positioning strategy like the one described here, i.e. only the initial reading of the GPS is used in the 'placement' of the temple, the most important aspects of the location sub-system are incremental changes, which affect distance resolution when walking, and stability, which affects the consistency and realism of the virtual world.

The GPS sub-system demonstrated adequate stability only for coarse positioning, with a drift in range of the order of ±2 m. This range of drift results in inconsistencies and noticeable virtual world 'shake' when the virtual elements are in close proximity of the user, as positional errors are more apparent at small distances. Distance resolution is of similar accuracy, affecting the representation of walking distances in the virtual world which, due to the drift, are not consistent with that of the real world.

The homegrown, low-cost DPGS solution [107, 233], based on the use of a laptop equipped with a Garmin 12XL GPS unit for a reference station and transmission of the correcting information over a wireless connection to the wearable, gave comparable results to the simple GPS solution. It is important to note that this approach is not as accurate as subscription-based DGPS, however it is cheaper to implement and does not require specialised receivers, only a wireless connection between the mobile and reference unit. Overall, the outcome of these experiments shows that the location sub-system, although accurate for coarse positioning, is not adequate for accurate real-time wearable AR as the drift of the GPS introduces observable, unrealistic movement of the virtual information in relation to the real environment.

The third set of experiments involved lab-based, quantitative investigations of the orientation subsystem. The HMD tracker was assessed both in terms of accuracy and stability. The orientation mechanism, proved fairly accurate: errors are within the required levels ($< 0.5^\circ$) and stability is fairly good, when the head is stable, with fluctuations within $\pm 0.2^\circ$.

The author's prototype was also evaluated by 40 users of ages ranging from 19 to 56 years, males and females, and of various levels of computer experience. The assessment attempted to investigate the usability of the system by questioning participants on certain features of the wearable after they used it on normal operating conditions. The survey was based on the use of a questionnaire, comprising of two sections, an open-ended question set and a more detailed, fixed choice question set.

The results demonstrated that, overall, participants felt that the system's usability was good. Strong points of the system proved to be the comfort of the vest and HMD, the sense of depth in the virtual world, the stability when users remained still and moved only their heads, and the overall impression it left. However, a series of shortcomings were identified, that affect the overall impression, the sense of presence and therefore the system's usability. The main problems were the narrow field-of-view of the HMD, its poor brightness, the fluctuations and instability of the head-tracker when participants walked, and the limited realism of the virtual world.

It is apparent that a replacing the HMD with one that has a wider FOV and adequate brightness for outdoor use can improve the visual output quality. Head-tracking stability can be improved either with the use of sensors designed for mobile use, or perhaps with the use of Kalman filters [112, 133]. Increasing the realism of the virtual world can be done with the use of stereo-rendering and anti-aliasing [104], shadows and textures [245, 228], while it may be possible to maintain rendering speed by using optimisations such as display lists [203] in addition to those already implemented. However, maintaining adequate rendering speed with such features enabled is also dependant on the hardware acceleration of the underlying platform, and is therefore dependent on the availability and capabilities of the software driver. This observation demonstrates that wearable AR involves a series of trade-offs between graphics performance, hardware platform features, battery-power, size and weight.

Chapter 7

Conclusions

7.1 The aim

The aim of this research was to develop an untethered, outdoor, mobile, augmented-reality tour guide using a wearable computer. The tour guide was to run *in situ*, in the area of the Gosbecks Archaeological Park at the outskirts of Colchester.

Proof of concept had been established from previous research efforts, most notably the Touring Machine [71]. More recent research on wearable AR has followed a similar paradigm, for example the Tinmith project [167, 169, 170, 171]. However these prototypes are bulky, heavy and expensive. Moreover, their purpose is to augment the real world using a limited set of 3D reconstructions relying, in addition, on audio and 2D annotations [71], or to investigate interface mechanisms for wearable AR [167].

The embedded electronic platforms usually employed in building wearable prototypes do not have hardware-accelerated graphic controllers; and commercial wearables are mostly low-power systems not able to cope with 3D reconstructions, as pinpointed by Zhong *et al.* [256]. The lack of fast rendering capability comes in addition to the intrinsic problems of registration and system delay [19] encountered in AR. In wearable AR, problems are amplified by the requirement for low-power, low-weight and small-size configurations. To address these challenges, researchers in the past opted for systems built around commercial laptops [71, 167] resulting in systems that, despite being powerful enough, suffer from the aforementioned problems. The author believes that these observations demonstrate one of the major challenges of untethered AR, that of overcoming the rendering bottleneck of wearable computers and achieving realistic 3D environments while maintaining adequate frame rates, a claim supported by Baillot *et al.*

[24], Gleue *et al.* [80] and Piekarsky [165, 169]. The problem is more severe in wearable AR than normal AR applications, where rendering is not so much of an issue, as stated by Azuma *et al.* [19].

Through this research, it was possible to explore to what extent wearable AR prototypes can be made more practical, ergonomic — i.e. with considerably less weight and less bulky than earlier prototypes — and low-cost, based on the use of off-the-shelf components. It further aimed to explore the development of a wearable AR framework that exploits rendering speed-ups, like those used in the gaming industry, is independent of the 3D models, and provides comparable, if not improved, performance compared to earlier research efforts. Last but not least, a user assessment was undertaken in order to evaluate the usability of the Romulus wearable and the Gosbecks tour guide.

7.2 Contributions made by this research

To summarise, the contributions of this research are:

Application: The development of a novel wearable AR system that allows inexperienced users to experience *in situ* the 3D architectural reconstruction of an archæological site.

Hardware: The design and construction of a novel wearable computer, named Romulus (Chapter 4), that has adequate rendering performance (10–15 fps when rendering a 30,000 polygon architectural model) and powered from a set of batteries for two hours. Romulus is a low-cost wearable computer based on a mini-ITX motherboard. It uses a Virtual I/O I-glasses HMD, with a head tracker and a handheld Garmin 12XL GPS unit to deduce orientation and location information respectively. It weighs 4.325 kg and is integrated into a vest for comfort.

Cost: The implementation of the system was based on off-the-shelf components of low cost (motherboard cost $< \pounds 90$), allowing easy and inexpensive replication of the main unit of the wearable. The HMD and the handheld GPS units that were used were available from previous research efforts, thus reducing implementation cost even further.

Software: The design and implementation of a novel wearable AR framework (Chapter 5), written using OpenGL, that renders a 3D architectural reconstruction at the aforementioned frame rates, exploiting DRI and employing optimisations such as a scene-graph, view frustum culling and levels-of-detail. The orientation and location sub-systems manipulate the user's view of the virtual world, with comparable accuracy to earlier prototypes. The framework is independent of the 3D model which be easily replaced with alternative scenes.

Assessment: A novel post-experiment questionnaire was designed and used in order to assess *in situ* the usability of the system that was developed (Chapter 6). Topics explored include the visual output quality, the simulation fidelity, the comfort when worn and the sense of presence induced. The questionnaire can be used in further assessments to aid the development of future wearable AR systems.

7.3 Principal conclusions

Through the implementation of Romulus and the Gosbecks tour guide it became apparent that it is possible to produce wearable AR prototypes that employ 3D graphics that are not complex, cumbersome and expensive while they maintain the rendering performance, and location and orientation determination accuracy of earlier prototypes. Romulus, built at low cost from off-the-shelf components, is an improvement over earlier prototypes in terms of ergonomics and comfort as it much lighter, small enough to fit in a modified photographers vest, and easy to construct and replicate. By exploiting hardware acceleration and software optimisation techniques the performance of the system overcame the minimum acceptable level of 10 fps while rendering a scene more complicated than those rendered by earlier prototypes.

The Romulus wearable and the Gosbecks tour guide were evaluated in a series of experiments investigating the system's accuracy and stability in determining location and orientation and an *in situ* user assessment, presented in Chapter 6, attempting to derive user feedback on the usability of the system in normal operating conditions. This makes Romulus one of the few wearable AR systems that employ 3D architectural reconstructions that have been assessed by users lacking experience in VR/AR.

The first group of experiments evaluated the real-time accuracy and positional stability of the standard and differential GPS mechanisms. The test results showed that GPS introduces a significant drift with time, easily observable in real-time. Virtual objects seem to move within about two metres, unrealistically, while they should remain stable. In practice, this means that users may observe inconsistencies when walking between virtual and real elements coexisting in the scene. Arguably, although GPS can give coarse positioning, its real-time accuracy and stability are not adequate for achieving the aforementioned accuracy levels and therefore not appropriate for precise wearable AR for 3D architectural reconstruction. Alternative sensors with higher positional accuracy need to be explored.

The second group of experiments evaluated the real-time accuracy and stability of the Virtual I/O head-tracker. This seems to perform satisfactorily, being accurate to $\pm 0.2°$. Nevertheless, it is sensitive to vibrations introduced by the user while walking, resulting in unrealistic vibration of virtual objects and to delays occurring from rapid head movements.

The third group of experiments involved a questionnaire-based user assessment of the system in terms of its ergonomics, quality of the visual output, simulation fidelity, sense of presence and overall impression. The collected data were used to form an overall evaluation of the Romulus-Gosbecks Tour guide system. The user assessment demonstrated that, overall, participants felt that the system's usability was good. The comfort of the wearable, the sense of depth in the virtual world, the stability when users remained still and moved only their heads, and the overall impression of the users were the strong points of the system. However, the narrow field-of-view of the HMD, the poor brightness of the display, the fluctuations and instability of the head-tracker when participants walked and the limited realism of the virtual world were considered as deficiencies.

An important conclusion of this research is that the evaluation of advanced human-computer interaction systems, such as AR-enabled wearables, in normal operational conditions by users is essential. Mid- and post-development phase usability assessments can identify problems with prototype designs and allow researchers to refine their implementations. As wearables become lighter and easier to use, practical investigations will become more successful and informative.

7.4 Discussion

Through this research it became apparent that the current breed of COTS embedded boards, can be used to construct fairly high-processing-power, low-cost, low-weight and low-energy-consumption wearable AR prototypes. In addition, constructing a wearable based on such a board is as easy as assembling a desktop PC. The mini-ITX platform has drawn extensive attention from developers, though not for wearables or AR, providing a widespread reference community.

The author anticipates that in the future a series of analogous features will be available in a number of embedded platforms. A smaller flavour of the mini-ITX boards with comparable features, the nano-ITX,[1] is expected to become popular with developers due to its small size (120mm×120mm) and low-consumption 1 GHz C3 CPU. Graphics card vendors are trying to introduce small form-factor GPUs, such as Nvidia's Quadro FX Go[2] for laptop graphics cards, that allow usage with embedded boards. Such processors are appearing in military wearable prototypes such as the Quantum3D Thermite wearable system presented in Chapter 2. Generally, the computing industry is beginning to focus on multimedia-rich mobile devices and embedded systems. It is only a matter of time until these GPUs appear on commercially available motherboards, enabling them to be exploited by wearable AR researchers to produce more powerful, yet low-cost, mobile/wearable AR systems while achieving practical and sensible ergonomic standards.

Virtual worlds that are augmented using wearable AR systems need to be hierarchically-structured into scene-graphs. By arranging the virtual scene spatially and separating it into nodes (groups of spatially adjacent objects), the application can decide on the elements that need to be rendered based on the user's field of view, a technique knows as view-frustum-culling. Levels-of-detail techniques can improve performance even further. Such optimisations are essential for achieving adequate frame rates on current hardware. Particularly for wearable AR, where the configurations are of limited capabilities, any technique that limits the amount of polygons to be rendered per frame is valuable. The author believes that such techniques and practices, employed in modern games, can be beneficial to wearable AR. The gaming industry is one of the fastest-growing sectors of computing, with continual advancements in software rendering techniques, physics simulators and graphics toolkits, also having impact on the development of graphical processors.

Overall system delay of wearable AR systems is also dependant on the sensors used. As has been mentioned in Chapter 1, system delay is the sum of the sensor, processing and rendering delays. In order to increase the simulation fidelity of the system, apart from increasing rendering speed, minimising the sensor delay is important. Faster sensors will detect movement and head rotations with higher fidelity and introduce smaller delays.

[1] http://www.viaembedded.com/

[2] http://www.nvidia.com/

7.5 Further work

7.5.1 Possible improvements

A series of improvements could be made to the system in order to remedy the problems identified on the author's platform and to result in an improved wearable AR system that exhibits better interface quality, accuracy, stability and ergonomics.

Interfaces: As was identified in Chapter 6, one of the areas for improvement of Romulus is the HMD. The Virtual I/O i-glasses has a narrow field-of-view and its brightness and contrast are poor for outdoor use. Cost factors did not allow replacement to take place in this research; however the author considers it is prudent to replace it with a newer, state-of-the-art HMD in order to increase the usability of the system. Also, it is wise to include a simple interface mechanism for application control, with buttons for resetting, restarting and pausing the application.

Vest and hardware improvements: The vest should be replaced from one made of heavy-duty fabric to increase sturdiness. Lighter battery packs could improve overall weight and comfort. Replacement of the hard disk drive by a compact flash card would reduce power consumption and weight even further.

Sensory modalities: The author believes that GPS accuracy is nowhere near good enough for precise, real-time wearable AR and that alternative sensory modalities need to be explored. These could be radio-frequency (RF) beacons, similar to the MIT Cricket [177] and Locust [210] systems; or computer vision tracking. Either or both could be used in conjunction with GPS in multi-modal, hybrid sensor arrays [22, 30, 157, 190]. Orientation tracking seems to be more satisfactory in terms of accuracy, however further improvement is required to increase stability and reduce delays. A popular method is the use of Kalman filters in the estimation of yaw, pitch and roll in the presence of noise [20, 133, 179, 243]. This can be fairly easily incorporated in the Gosbecks tour guide software, replacing the simple filter currently used (see Chapter 5).

Software improvements: A large percentage of users would prefer a more realistic virtual world (see Chapter 6). The use of textures, shading, stereo-rendering and anti-aliasing could improve realism. However these are dependant to a large extent on the hardware platform and its graphics capabilities. These features can be added in the current system as support for them becomes available in the driver of the CLE266 chipset used in Romulus. Other hardware platforms, like the aforementioned Nvidia chipsets likely to be available in the future, may be more capable and incorporate the required features. To maintain adequate frame rate with the aforementioned improvements, the application requires further optimisations such as the use of display lists [203] that encapsulate the rendering calls of virtual objects that will be drawn repeatedly (like columns).

Additional assessment scenarios: The independence of the developed application from the Gosbecks model allows additional architectural reconstructions to be used. The latter may have further rendering requirements such as occlusion, not required in this example, that may allow discovery of other types of deficiencies in the existing sensors and interfaces, or may be used to evaluate modalities added in subsequent systems. Assessment in different scenarios might generate further topics for research and will allow further insight on the usability of wearable AR system.

7.6 The future of wearable AR

This thesis described the author's attempt to investigate the practicality of wearable AR that uses 3D graphics, through a sensible, hands-on approach. The inherent problems of wearable computing — and AR in general — impose a number of challenges that are difficult to address with current technology. Although this research explored to some extent the trade-offs between rendering performance, power consumption, size, cost and complexity there are a number of unresolved issues that affect the performance of wearable AR systems.

The accuracy of current registration and location determination methodologies is still lacking. Precise wearable AR requires more accurate sensing, particularly of position. Orientation sensors seem to be adequate, though the one used here was susceptible to instabilities. Indeed, the overall effects of mis-registration errors may not be overcome easily.

The use of commercial HMDs for output introduces a number of further problems related to their optics. Focus mismatch can cause eye strain and ocular discomfort, whereas trade-offs between resolution, field-of-view and the weight of optics result in cumbersome, heavy systems. The requirement for easy HMD calibration is also an issue that affects the practicality of most AR applications in general, therefore affecting wearable AR as well. Further research in the devices used in HMDs is also required. Overall, wearable AR is still in its infancy, facing a series of challenges that require considerable research to overcome. The author believes that researchers should constantly try to exploit technological advancements in order to address those challenging issues, aiming to improve the sense of presence their systems introduce.

Appendix A

The Gosbecks archæological site

The invasions of Julius Cæsar of Britain in 55 and 54 BC were supposedly to avenge the murder of the king of the Trinovantes, who inhabited modern Essex and southern Suffolk, by his rival, the king of modern Hertfordshire. After Cæsar's departure, the two tribes were united into a single royal house which established its capital at *Camulodunum* ("fortress of the war god"), to the south-west of modern Colchester at a place now known as the Gosbecks Archæological Park. Camulodunum exported grain, meat, hides, hunting dogs and even oysters to the Roman world and reached its zenith under the reign of Cunobelin (Shakespeare's Cymbeline).

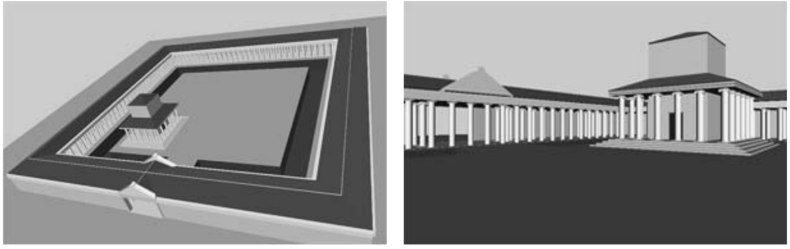
Shortly after Cunobelin's death, Claudius decided to bolster his reputation by conquering Britain. In 43 AD, his 40,000-strong army landed in Kent and fought its way northwards, stopping just short of Camulodunum. The Roman commander sent for Claudius who travelled from Rome to lead his troops into the city in person. In 49 AD, the first *colonia* (colony) of the new province of Britannia was established next to Camulodunum, on the site of modern Colchester. The Gosbecks area was also developed further with a 5,000-seat theatre, the largest in Britain, and a temple complex.

The colonists' treatment of the locals was apparently not good. In particular, when Prasutagus, king of the Iceni, died in 60 AD, his wife Boudicca ("Bodicea") was beaten by Roman officials and her daughters raped. The Iceni rose in fury and destroyed the colony, burning its temple (the site of the modern castle) where many colonists had taken refuge. Boudicca went on to ravage London and *Verulamium* (at the edge of modern St. Albans) before being apprehended and killed. The colony at Colchester was rebuilt, this time with the extensive walls that remain today.

Following the end of Roman rule circa 450 AD, future development was centred within the city walls, in modern Colchester. The Gosbecks site proved an excellent source of building materials for local inhabitants — so much so that all that effectively remains today are foundations, and these are below the ground level. Indeed, the only way that a visitor to Gosbecks is aware of the scale and layout of the former buildings is by white lines painted on the ground and accompanying signage.

Temple architecture

Roman architecture was based on the principles of the earlier, more elegant Greek building styles; however, their approach was more formulaic to enable faster construction and employ less skilled artisans. Fortunately, a guide to building design due to Vetruvius [151] has survived from antiquity. With this guide, the author was able to write an OpenGL 3D model (see Chapter 5) of Roman buildings from a few key measurements such as the number of columns and their diameters, based on earlier VRML models due to Christine Clark.



(a) The Gosbecks Temple from above (b) Typical user's view

Figure A.1: Views of the Gosbecks temple grabbed from the authors' VRML models

The temple complex comprises a square-shaped portico with an outer wall, an outer circuit of Doric columns and an inner circuit of Ionic columns, all being covered by a tiled roof. The entrance faces roughly east. Within the portico is a ditch and, in the south-east of that, the main temple (fig. A.1). The temple covers an area slightly less than 7000 square metres, resulting in a sizable 3D model with around 250 columns (both Doric and Ionic). The complexity of the models is such that, in order to render properly, software optimisations were required in order to achieve satisfactory frame rates (see Chapter 5).

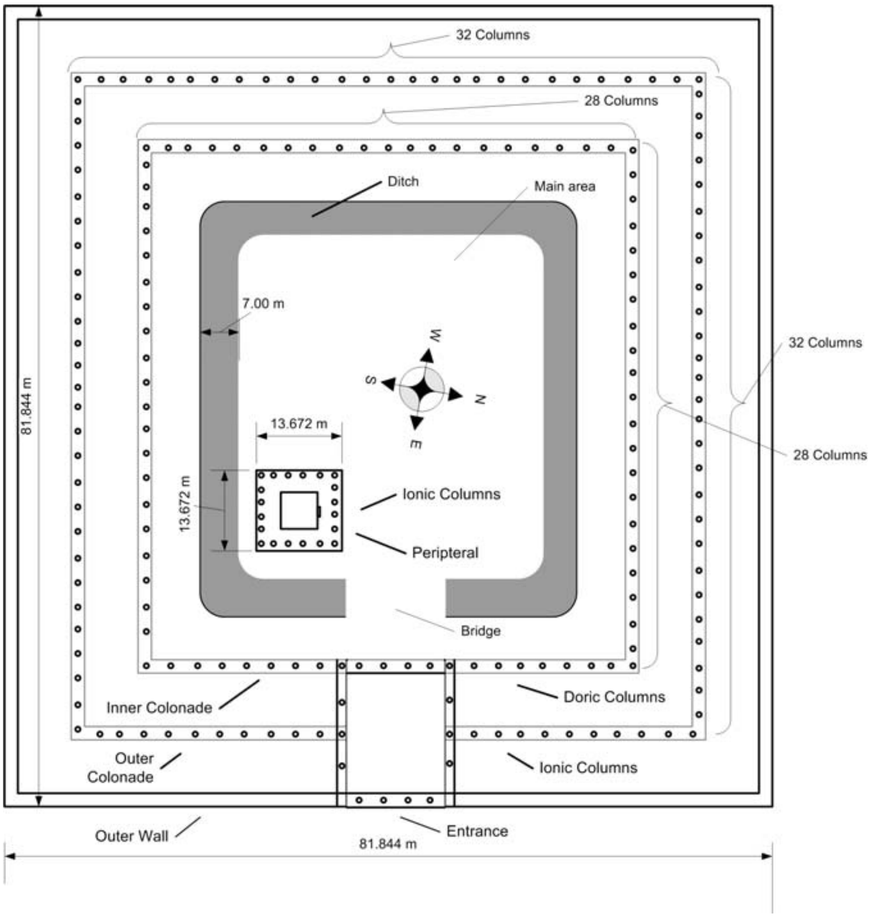


Figure A.2: The Gosbecks main Temple (not to scale)

Appendix B

The Remus Wearable Computer

Remus was built during the author's first research year, in 2000–2001. The aim was to construct a simple wearable, not novel, system and use it for *in situ*, outdoor testing. Most boards — of various form factors (Chapter 2) — available at that period were comparable in terms of performance and interfaces. The choice of the PC/104-based 'Tin Lizzy' architecture was done because of the simplicity and popularity of the design, plus the availability of spare PC/104 cards in the author's laboratory. The popularity of PC/104 boards was such that a number of configurations were possible. The main board chosen was a Digital Logic[1] MSM5SEV Pentium-based PC/104 motherboard that offered the potential of upgrade to a Pentium-III. The board was the fastest option available at that period and seemed more than adequate for a basic system. Alternatives were mostly PC/104 variants of lower specification and without the upgrade potential of the MSM5SEV.

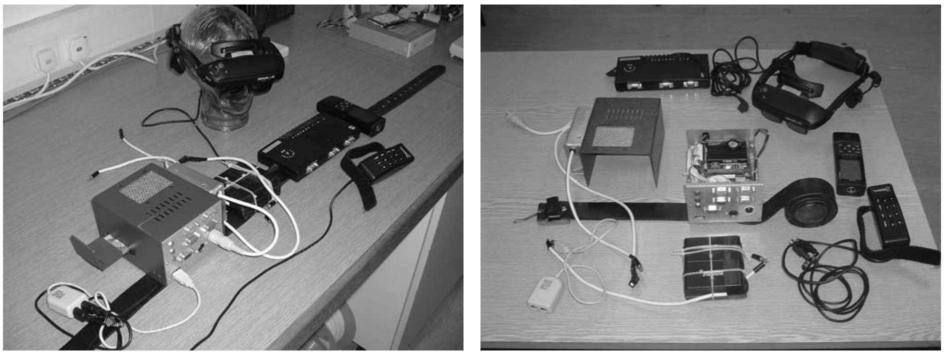
Remus was eventually constructed from three PC/104 boards, a DC-to-DC converter and used for output the Virtual I/O HMD and a Twiddler2 for input. The other two boards apart from the motherboard were a Eurotech[2] soundcard with two extra serial ports and a Jumptec PC Card adapter[3]. The extra serial ports were to be used for the GPS unit. The initial design required 3 serial ports for the GPS, HMD tracker and Twiddler; yet, with the introduction of the Twiddler2, only two serial ports were eventually used. The device was mounted on a belt, on the right hip of the user. A steel case was constructed to include the PC/104 stack and a small expansion board accommodating connectors for the IrDA, Ethernet and USB interfaces. A separate enclosure held the DC-to-DC converter circuit. A dual PC Card slot was machined into

[1] `http://www.digitallogic.ch/`

[2] `http://www.eurotech.it/`

[3] `http://www.jumptec.de/`

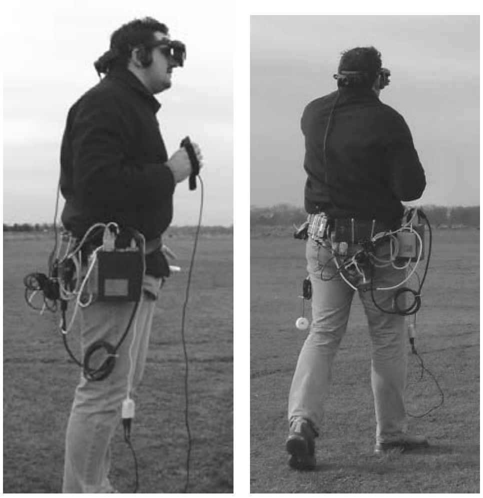
the side of the case. The wearable was completed with the Virtual i/O HMD driver unit and a set of camcorder batteries. A Garmin GPS12XL GPS unit provides location information and the HMD tracker provides information on the user's direction of field of view.



(a) The Remus wearable assembled

(b) The Remus components

Figure B.1: The Remus wearable



(a)

(b)

Figure B.2: The author wearing Remus

The ‘Remus’ wearable computer specification

Main Components	
CPU board	Digital Logic’s MSM5SEV PC/104CPU board
Soundcard	Eurotech’s I/O SB PC/104 Soundcard
PC Card adapter	Jumptec’s PC/104-to-PC Card adapter
I/O Devices	
Input	HandyKey Twiddler2
Output	Virtual IO Glasses/Micro-Optical SV-9
GPS	Garmin GPS12XL GPS Unit
WLAN	Wireless LAN PC card
Specification	
CPU	Pentium 266 MHz CPU (Upgradeable to a PIII 700 MHz)
Memory	64 MB of RAM (Upgreadable to 128)
Chipset	On Board 69000 VGA Output – Input
Sound	Soundcard
Ports	4 Serial Ports, 2 Parallel, 1 USB, PS/2, IrDA
Network	Ethernet 10/100, Wireless LAN (802.11)
Expansion	PC card Type II/III slot
Storage	3 GB HDD – Toshiba
Power	12V DC input, 5V@25W, 9V@1W output
Batteries	2X Duracel DR11 NiMH
Operating System	RedHat 7.2 with Mesa3D graphics library V.3.4.2-10

Appendix C

Romulus schematics

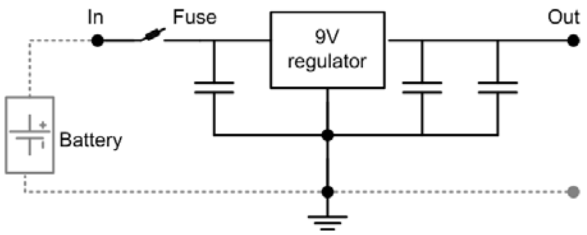


Figure C.1: The 9 Volt regulator circuit

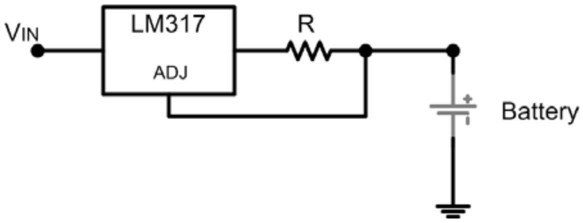
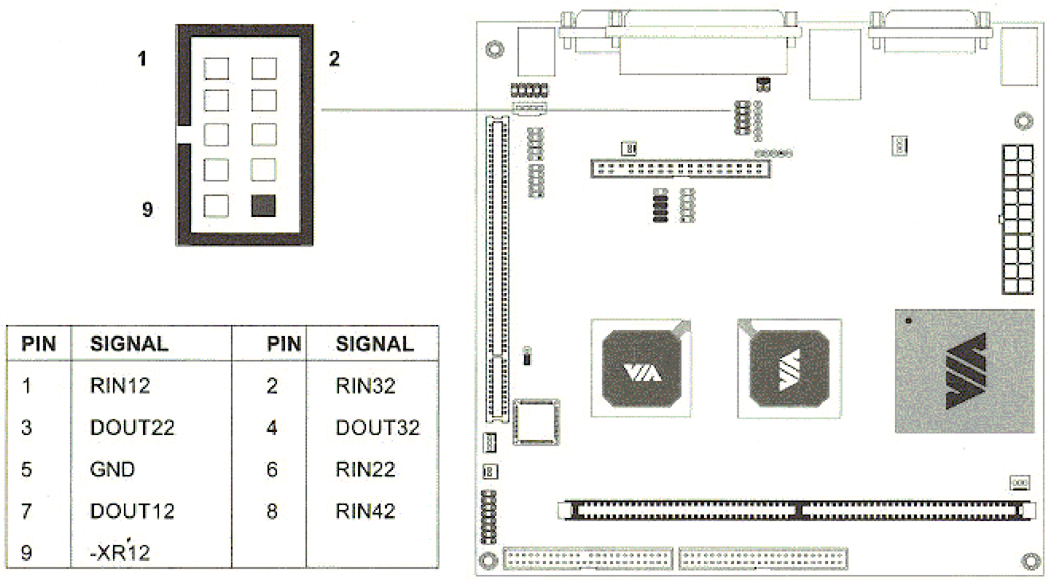


Figure C.2: The batteries' charger schematic

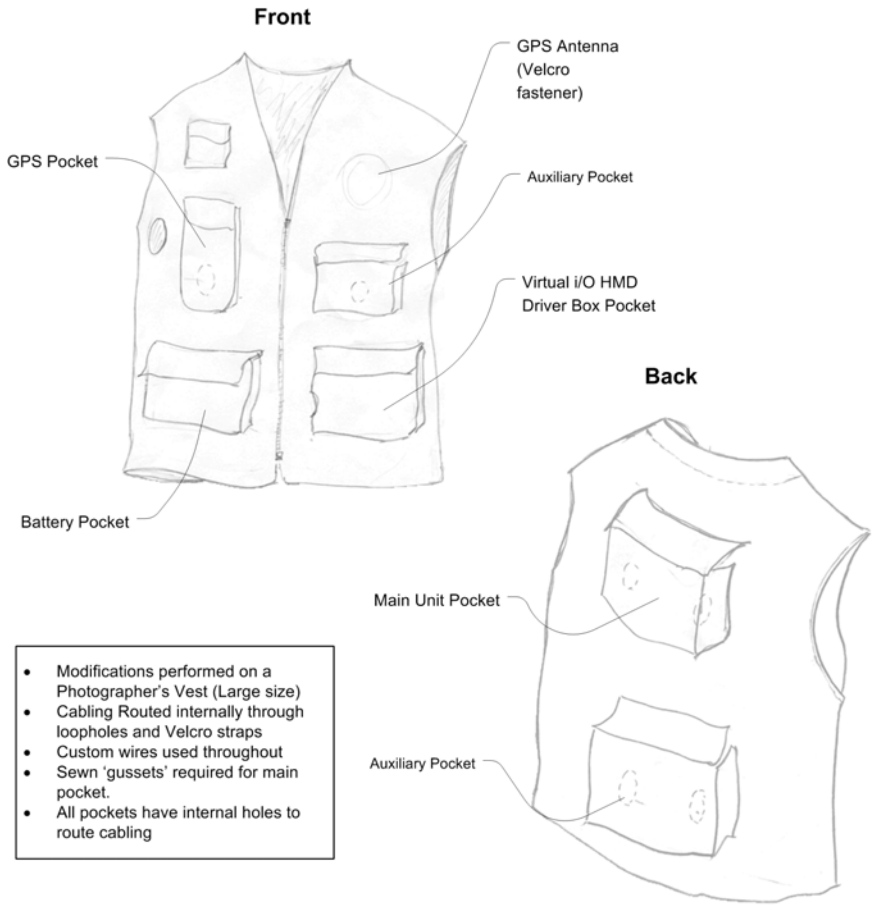


PIN	SIGNAL	PIN	SIGNAL
1	RIN12	2	RIN32
3	DOUT22	4	DOUT32
5	GND	6	RIN22
7	DOUT12	8	RIN42
9	-XR12		

Figure C.3: The Mini-ITX second serial port internal connector (VIA Technologies Inc.)

Appendix D

Romulus Vest Prototype Drawings



- Modifications performed on a Photographer's Vest (Large size)
- Cabling Routed internally through loopholes and Velcro straps
- Custom wires used throughout
- Sewn 'gussets' required for main pocket.
- All pockets have internal holes to route cabling

Figure D.1: Romulus Vest Prototype

Appendix E

Romulus Case Diagram

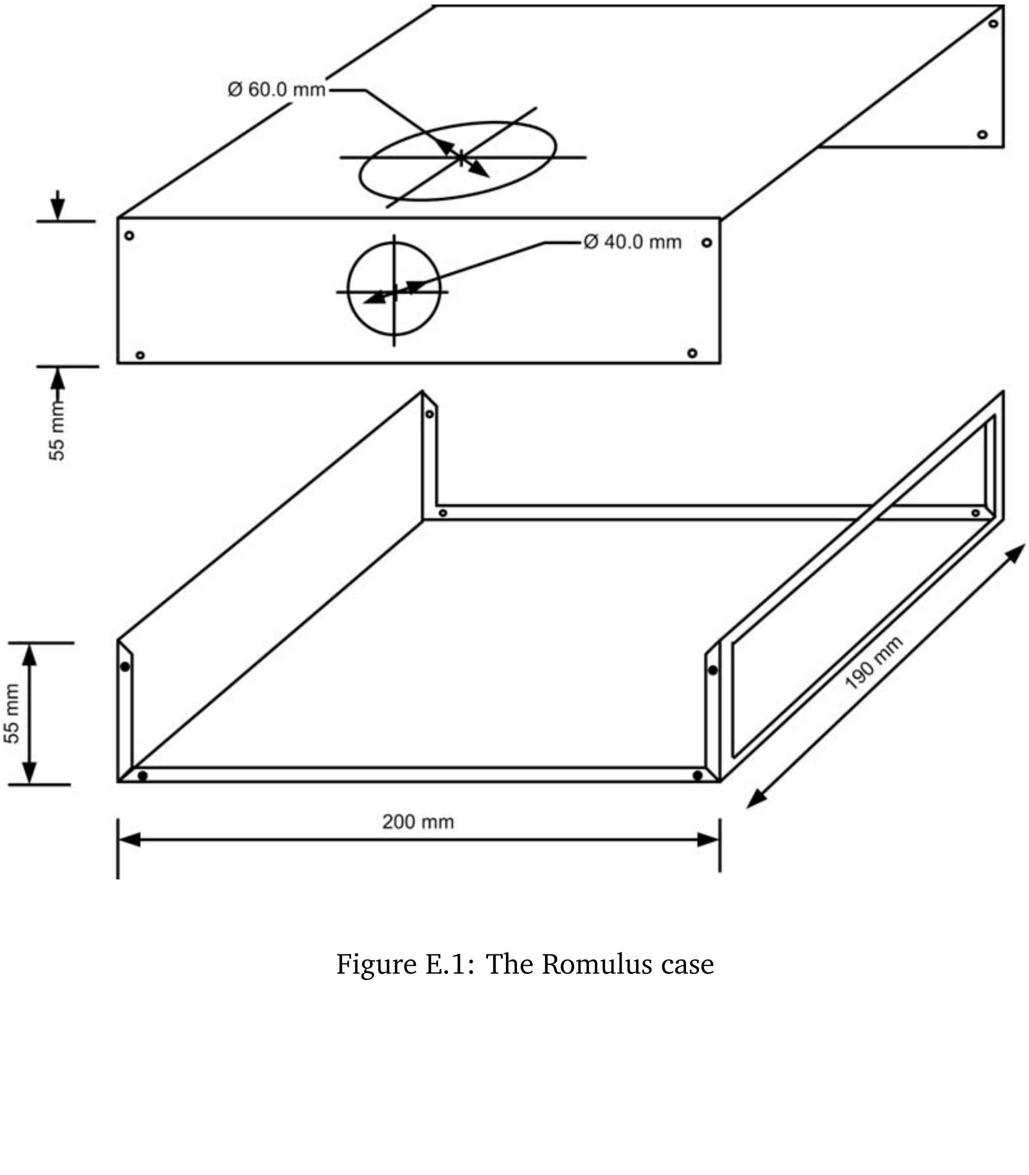


Figure E.1: The Romulus case

Appendix F

Custom visor for Virtual I/O HMD

Three visors were constructed and their specification is presented below:

Visor	Density	Reduction by f-stops
1	0.3	1
2	0.6	2
3	0.9	3

Table F.1: Filter specifications

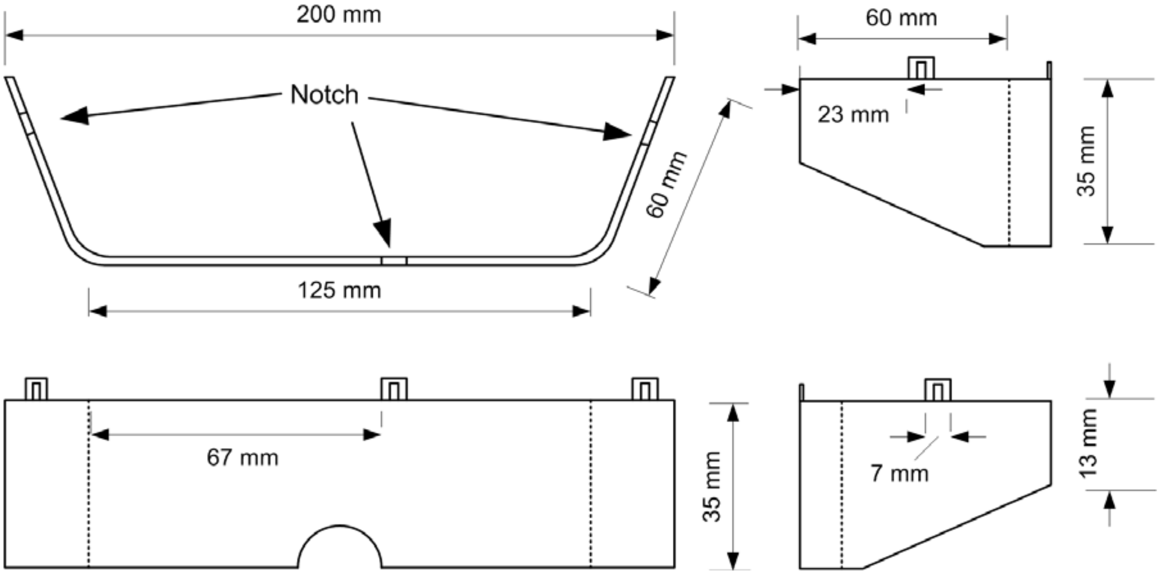
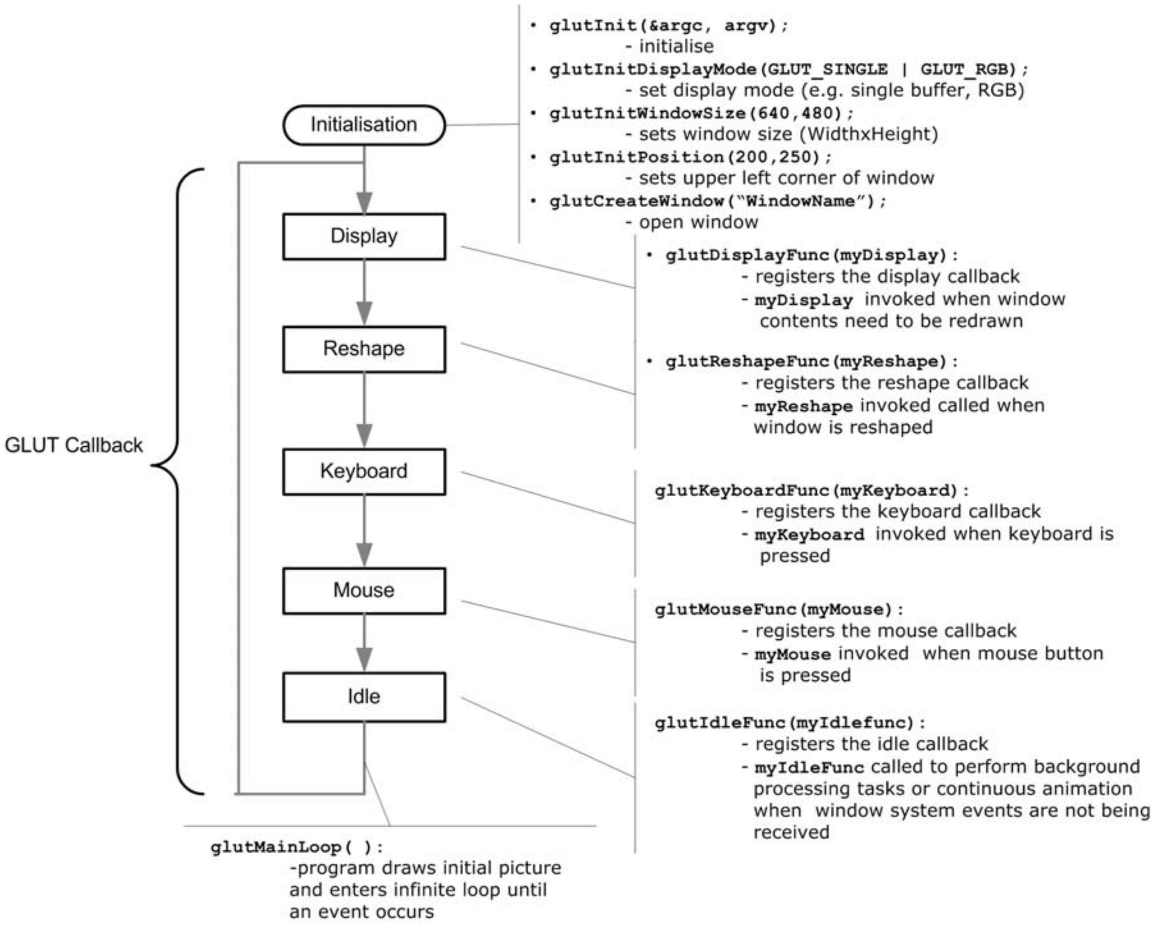


Figure F.1: Custom visor dimensions

Appendix G

The Glut Callback loop



- **glutInit(&argc, argv);**
  - initialise
- **glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);**
  - set display mode (e.g. single buffer, RGB)
- **glutInitWindowSize(640,480);**
  - sets window size (WidthxHeight)
- **glutInitPosition(200,250);**
  - sets upper left corner of window
- **glutCreateWindow("WindowName");**
  - open window

- **glutDisplayFunc(myDisplay):**
  - registers the display callback
  - **myDisplay** invoked when window contents need to be redrawn
- **glutReshapeFunc(myReshape):**
  - registers the reshape callback
  - **myReshape** invoked called when window is reshaped

glutKeyboardFunc(myKeyboard):
- registers the keyboard callback
- **myKeyboard** invoked when keyboard is pressed

glutMouseFunc(myMouse):
- registers the mouse callback
- **myMouse** invoked when mouse button is pressed

glutIdleFunc(myIdlefunc):
- registers the idle callback
- **myIdleFunc** called to perform background processing tasks or continuous animation when window system events are not being received

glutMainLoop( ):
-program draws initial picture and enters infinite loop until an event occurs

Figure G.1: The GLUT callback loop

Appendix H

Examples of Stereo Rendering using Glut

These examples are based on Paul Burke's tutorials on stereo rendering using OpenGL and Glut, available at:

`http://astronomy.swin.edu.au/∼pbourke/opengl/stereogl/`

Rotation method

```
 glutInit(&argc,argv);
   if (!stereo)
      glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
   else
      glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH | GLUT_STEREO);


   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   gluPerspective(camera.aperture,screenwidth/(double)screenheight,0.1,10000.0);

   if (stereo) {

      CROSSPROD(camera.vd,camera.vu,right);
      Normalise(&right);
      right.x *= camera.eyesep / 2.0;
      right.y *= camera.eyesep / 2.0;
      right.z *= camera.eyesep / 2.0;

      glMatrixMode(GL_MODELVIEW);
      glDrawBuffer(GL_BACK_RIGHT);
      glLoadIdentity();
      gluLookAt(camera.vp.x + right.x,
                camera.vp.y + right.y,
                camera.vp.z + right.z,
                focus.x,focus.y,focus.z,
                camera.vu.x,camera.vu.y,camera.vu.z);
      MakeLighting();
      MakeGeometry();

      glMatrixMode(GL_MODELVIEW);
      glDrawBuffer(GL_BACK_LEFT);
      glLoadIdentity();

      gluLookAt(camera.vp.x - right.x,
                camera.vp.y - right.y,
                camera.vp.z - right.z,
                focus.x,focus.y,focus.z,
                camera.vu.x,camera.vu.y,camera.vu.z);
      MakeLighting();
      MakeGeometry();
   } else {
      glMatrixMode(GL_MODELVIEW);
      glDrawBuffer(GL_BACK_LEFT);
      glLoadIdentity();
      gluLookAt(camera.vp.x,
                camera.vp.y,
                camera.vp.z,
                focus.x,focus.y,focus.z,
                camera.vu.x,camera.vu.y,camera.vu.z);
      MakeLighting();
      MakeGeometry();
   }
   glutSwapBuffers();
```

Parallel Projection Algorithm

```
glutInit(&argc,argv);
if (!stereo)
   glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
else
   glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH | GLUT_STEREO);


ratio  = camera.screenwidth / (double)camera.screenheight;
radians = DTOR * camera.aperture / 2;
wd2     = near * tan(radians);
ndfl    = near / camera.focallength;


glDrawBuffer(GL_BACK_LEFT);
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
if (stereo) {
   glDrawBuffer(GL_BACK_RIGHT);
   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
}

if (stereo) {


   CROSSPROD(camera.vd,camera.vu,r);
   Normalise(&r);
   r.x *= camera.eyesep / 2.0;
   r.y *= camera.eyesep / 2.0;
   r.z *= camera.eyesep / 2.0;



   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   left  = - ratio * wd2 - 0.5 * camera.eyesep * ndfl;
   right =   ratio * wd2 - 0.5 * camera.eyesep * ndfl;
   top    =   wd2;
   bottom = - wd2;
   glFrustum(left,right,bottom,top,near,far);

   glMatrixMode(GL_MODELVIEW);
   glDrawBuffer(GL_BACK_RIGHT);
   glLoadIdentity();
   gluLookAt(camera.vp.x + r.x,camera.vp.y + r.y,camera.vp.z + r.z,
             camera.vp.x + r.x + camera.vd.x,
             camera.vp.y + r.y + camera.vd.y,
             camera.vp.z + r.z + camera.vd.z,
             camera.vu.x,camera.vu.y,camera.vu.z);
   MakeLighting();
   MakeGeometry();
```

```
      glMatrixMode(GL_PROJECTION);
      glLoadIdentity();
      left  = - ratio * wd2 + 0.5 * camera.eyesep * ndfl;
      right =   ratio * wd2 + 0.5 * camera.eyesep * ndfl;
      top    =   wd2;
      bottom = - wd2;
      glFrustum(left,right,bottom,top,near,far);
      glMatrixMode(GL_MODELVIEW);
      glDrawBuffer(GL_BACK_LEFT);
      glLoadIdentity();
      gluLookAt(camera.vp.x - r.x,camera.vp.y - r.y,camera.vp.z - r.z,
                camera.vp.x - r.x + camera.vd.x,
                camera.vp.y - r.y + camera.vd.y,
                camera.vp.z - r.z + camera.vd.z,
                camera.vu.x,camera.vu.y,camera.vu.z);
      MakeLighting();
      MakeGeometry();
   } else {
      glMatrixMode(GL_PROJECTION);
      glLoadIdentity();
      left  = - ratio * wd2;
      right =   ratio * wd2;
      top    =   wd2;
      bottom = - wd2;

      glFrustum(left,right,bottom,top,near,far);
      glMatrixMode(GL_MODELVIEW);
      glDrawBuffer(GL_BACK_LEFT);
      glLoadIdentity();
      gluLookAt(camera.vp.x,camera.vp.y,camera.vp.z,
                camera.vp.x + camera.vd.x,
                camera.vp.y + camera.vd.y,
                camera.vp.z + camera.vd.z,
                camera.vu.x,camera.vu.y,camera.vu.z);
      MakeLighting();
      MakeGeometry();
   }


   glutSwapBuffers();
```

Appendix I

Preliminary User Assessment Questionnaire

USER ASSESSMENT FOR AR ARCHITECTURES Test No:[ ]

Experiment Profile

Software Version:________________

Wearable Computer:________________

Date of Experiment:________________

User Profile

Age:________________

Gender:________________

Please mark your responses with an 'X' in the appropriate box.

	Strongly Agree	Agree	Neither agree nor disagree	Disagree	Strongly disagree
I am an experienced computer user					
I am familiar with 3D computer games					
I am a frequent player of 3D computer games					
I frequently visit archaeological sites					
I am generally interested in Archaeology					

Is this the first time you have used a Head Mounted Display? Yes □ No □

Ergonomics

The wearable computer was heavy while worn	□	□	□	□	□
The wearable computer was too bulky	□	□	□	□	□
It was uncomfortable to wear the computer system	□	□	□	□	□
It was uncomfortable to wear the HMD	□	□	□	□	□
I could not move freely while wearing the wearable computer	□	□	□	□	□
Wearing the HMD affected my head movement	□	□	□	□	□

Realism

Rotating the HMD results in accurate movement	□	□	□	□	□
Movement in to the virtual model is realistic	□	□	□	□	□
The model registers properly in the real environment	□	□	□	□	□
The rendering speed of the model is adequate	□	□	□	□	□

Visual Output Quality

The graphics of the model were detailed	□	□	□	□	□
The colours of the models were realistic	□	□	□	□	□
The HMD has adequate brightness	□	□	□	□	□
The HMD has adequate resolution	□	□	□	□	□

General Remarks

I was impressed by the application implementation	□	□	□	□	□
Wearing the wearable computer was fun	□	□	□	□	□
I would like to see a similar application in archaeological sites	□	□	□	□	□
I would be willing to pay for similar AR tours	□	□	□	□	□

I would like to see other similar wearable computing applications Yes □ No □

Such as: ____________________

What was your general impression of the AR Tour Guide? (`5 Best - 1 Worst`): □

Appendix J

User Assessment Introduction

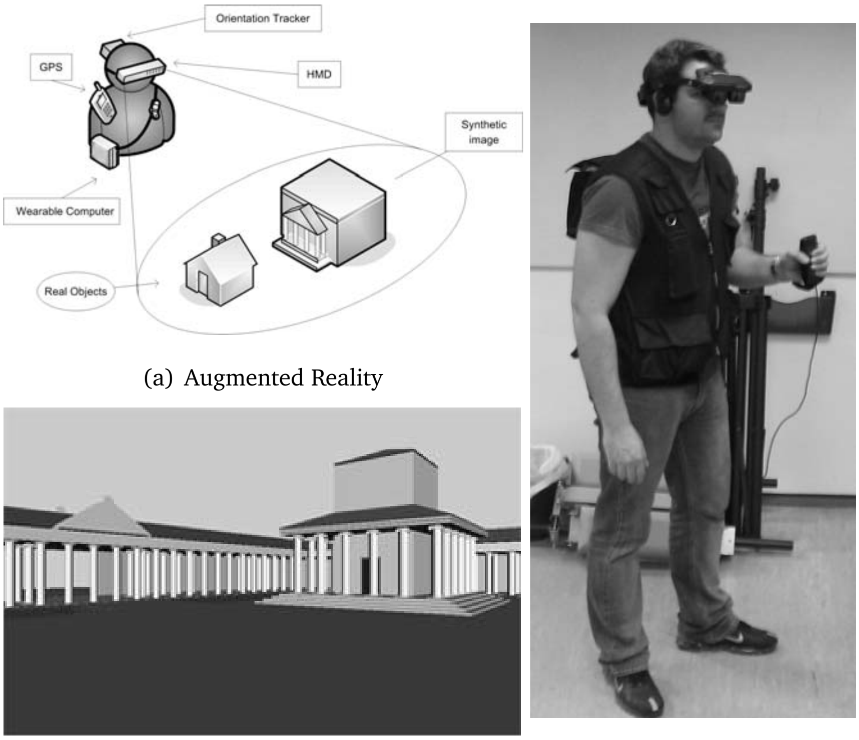
The following text was given to the survey participants to describe the experiment. The text was given in a single page, including the pictures on the next page.

User assessment for Wearable AR

The purpose of the experiment is to collect your feedback on the use of a **wearable computing augmented reality** experiment. A wearable computer is a *body-worn computer*, equipped with a *head-mounted display* (HMD) like the one shown in Fig.J.1(c)). Augmented Reality (AR) is similar to Virtual Reality (VR). In virtual reality you can only see a *computer-generated image*. In Augmented Reality you can see a virtual image *superimposed* on the real environment with the use of *see-through displays* (Fig.J.1(a)).

The project examined aims to provide an augmented reality tour of a *3D architectural model* (Fig.J.1(b)) of a Roman temple originally situated on the outskirts of Colchester. *While wearing the wearable computer you will be asked to walk on the football pitch in the University's campus*. The wearable computer is equipped with a GPS (global positioning system) location system which provides location information *i.e. where you are standing*. It is also equipped with an orientation detection mechanism which can detect head rotations *i.e. where you are looking at*. This information is used to update the view of a 3D model of the Roman temple. Therefore as you move on the field you see the 3D model 'on the field' *as if the real building was there*.

You will be asked to wear the wearable computer for 15 minutes, roam 'into' the temple by walking and turning your head as if you were in the real temple and, upon completion of the tour, to answer a 3-page questionnaire.



(a) Augmented Reality

(b) Temple 3D model

(c) The wearable Computer

Figure J.1: The wearable Computer

Appendix K

Finalised Questionnaire

Mobile Augmented Reality — User Assessment Questionnaire

For our use only

Questionnaire number . . . .

Date . . . .

Wearable computer . . . .

Software version . . . .

About You

Age . . . .

Gender . . . .

How would you rate your expertise with computers?

Expert □ Advanced □ Intermediate □ Novice □ No experience □

Is this the first time you have used a Head Mounted Display (HMD)?

Yes □ No □

Is this the first time you have used a body-worn (wearable) computer?

Yes □ No □

Comments

Q. 1. Do you have any comments about the head-mounted display?

Q. 2. Do you have any comments about the body-mounted computer?

Q. 3. Describe any unpleasant sensations you may have felt while using the system.

Q. 4. Any other comments?

To what extent did you experience the following:

		None	Slight	Moderate	Severe	Very Severe
Q. 5.	**tired eyes**	□	□	□	□	□
Q. 6.	**nausea**	□	□	□	□	□
Q. 7.	**headache**	□	□	□	□	□
Q. 8.	**dizziness**	□	□	□	□	□
Q. 9.	**disorientation**	□	□	□	□	□

Q. 10 How do you rate the comfort of the body-worn computer?

Excellent □ Good □ Adequate □ Poor □ Very poor □

Q. 11 How do you rate the comfort of the head-mounted display?

Very poor □ Poor □ Adequate □ Good □ Excellent □

Q. 12 How easy was it to find your way around in the virtual world?

Very easy □ Easy □ Average □ Difficult □ Very difficult □

Q. 13 Could you see the pixels (little squares) that made up the graphical objects?

Yes □ No □

Q. 14 If Yes, how noticeable were they?

Very noticeable □ Noticeable □ Not very noticeable □ Hardly Noticeable □ Barely visible □

Q. 15 Would you say that the display was:

Very bright □ Bright □ Average □ Dim □ Too dim □

Q. 16 Did you notice that your view of the virtual world was restricted by the edges of the display?

No, the view was perfectly natural □ No, the view was not noticeably restricted □ Yes, but the border was not very noticeable □ Yes, the border was quite noticeable □ Yes, the border was very restrictive □

Q. 17 How easy was it to combine the real and computer-generated scenes together?

Very difficult □ Difficult □ Average □ Easy □ Very easy □

Q. 18 How do you rate the realism of the graphical output?

Excellent □ Good □ Adequate □ Poor □ Very poor □

Q. 19 How do you rate your sense of depth in the virtual world while using the system?

Very poor □ Poor □ Adequate □ Good □ Excellent □

Q. 20 How do you rate your sense of scale in the virtual world while using the system?

Excellent □ Good □ Adequate □ Poor □ Very poor □

Q. 21 How stable was the virtual world when you stayed still?

Completely unstable □ Fairly unstable □ A little instability □ Quite stable □ Completely stable □

Q. 22 How stable was the virtual world as you moved your head?

Completely stable □ Quite stable □ A little instability □ Fairly unstable □ Completely unstable □

Q. 23 How stable was the virtual world as you walked around?

Completely unstable □ Fairly unstable □ A little instability □ Quite stable □ Completely stable □

Q. 24 How do you rate your sense of 'being there' while using the system?

Excellent □ Good □ Adequate □ Poor □ Very poor □

Q. 25 How do you rate your overall impression of the system?

Very poor □ Poor □ Adequate □ Good □ Excellent □

Thank you!

Bibliography

[1] R. Ackermann, M. Goertz, M. Karsten, and R. Steinmetz. Prototyping a PDA based Communication Appliance. In *Proceedings of International Conference on Software, Telecommunications and Computer Networks (SoftCOM'01), Split, Croatia*, 2001.

[2] C. Aimone and J. Fung. An EyeTap video-based featureless projective motion estimation assisted by gyroscopic tracking for wearable computer mediated reality. *Personal and Ubiquitous Computing 7.5*, pages 236–248, October 2003.

[3] C. Aimone, A. Marjan, and S. Mann. EyeTap video-based featureless projective motion estimation assisted by gyroscopic tracking. In *Proceedings of the Sixth International Symposium on Wearable Computers (ISWC '02)*. IEEE Computer Society, 2002.

[4] K. Akeley, S. Watt, A. Girshick, and M. Banks. A stereo display prototype with multiple focal distances. In *ACM Transaction on Graphics*, 2004.

[5] D. Alais and R. Blake. Grouping visual features during binocular rivalry. *Vision Res*, 39(26):44341–4353, October 1999.

[6] D. G. Aliaga, J. Cohen, A. Wilson, E. Baker, H. Zhang, C. Erikson, K. E. Hoff III, T. Hudson, W. Sturzlinger, R. Bastos, M. C. Whitton, F. P. Brooks Jr., and D. Manocha. MMR: an interactive massive model rendering system using geometric and image-based acceleration. In *Symposium on Interactive 3D Graphics*, 1999.

[7] A. Watt amd M. Watt. *Advanced Animation and Rendering Techniques: Theory and Practice*. Addison-Wesley, 1st edition, October 1992.

[8] S. L. Ames, J. S. Wolffgohn, and N. A. McBrien. The development of a symptom questionnaire for assessing Virtual Reality viewing using a head-mounted display. *Optometry and Vision Science*, 82(3):168–176, March 2005.

[9] C. H. Amon, E. R. Egan, A. Smailagic, and D. P. Siewiorek. Thermal management and concurrent system design of a wearable multicomputer. *Components, Packaging, and Manufacturing Technology, Part A, IEEE Transactions*, 20(2):128–137, June 1997.

[10] U. Anliker, P. Lukoxocz, G. Troester, S. J. Schwartz, and R. W. DeVaul. The wearARM: Modular, high performance, low power computing platform designed for integration into everyday clothing. In *Proceedings of the Fifth International Symposium on Wearable Computers (ISWC '01)*. IEEE Computer Society, 2001.

[11] American Psychological Association (APA). *Publication Manual of the American Psychological Association*. American Psychological Association (APA), 5th edition, June 2001.

[12] J. Ardo and P. Pilesjo. On the accuracy of the global positioning system — a test using a hand-held receiver. *International Journal of Remote Sensing*, 13:3229–3233, November 1992.

[13] D. Ashbrook and T. Starner. Learning significant locations and predicting user movement with GPS. In *Proceedings of the Sixth International Symposium on Wearable Computers (ISWC '02)*. IEEE Computer Society, 2002.

[14] P. J. Atkin. Parallel processing for virtual reality. University of Edinburgh, 1993. Parallel Processing for Graphics and Scientific Visualisation.

[15] S. Aukstakalnis, D. Blatner, and Stephen F. Roth. *Silicon Mirage — The Art and Science of Virtual Reality*. Peachpit Press, 1992.

[16] B. Avery, B.H. Thomas, J. Velikovsky, and W. Piekarski. Outdoor augmented reality gaming on five dollars a day. In *Proceedings of the 6th Australasian User Interface Conference (AUIC'05)*, January 2005.

[17] R. Azuma. A survey of augmented reality. In *In Computer Graphics (SIGGRAPH'95) Proceedings, Course Notes 9: Developing Advanced Virtual Reality Applications)*, 1995.

[18] R. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–385, 1997.

[19] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre. Recent advances in augmented reality. *IEEE Computer Graphics and Applications*, 21(6):34–47, 2001.

[20] R. Azuma and G. Bishop. Improving static and dynamic registration in an optical see-through HMD. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques (SIGGRAPH '94)*. ACM Press, 1994.

[21] R. T. Azuma. *Predictive Tacking for Augmented Reality*. PhD thesis, University of North Carolina, 1995.

[22] R. T. Azuma, B. R. Hoff, III H. E. Neely, R. Sarfaty, M. J. Daily, G. Bishop, L. Vicci, G. Welch, U. Neumann, S. You, R. Nichols, and J. Cannon. Making augmented reality work outdoors requires hybrid tracking. In *Proceedings of the international workshop on Augmented reality : placing artificial objects in real scenes (IWAR '98)*, 1999.

[23] Y. Baillot, D. Brown, and S. Julier. Authoring of physical models using mobile computers. In *Proceedings of the Fifth International Symposium on Wearable Computers (ISWC '01)*. IEEE Computer Society, 2001.

[24] Y. Baillot, E. Gagas, T. Ḧ(o)llerer, S. Julier, and S. Feiner. Wearable 3D graphics for augmented reality: A case study of two experimental backpack computers. Technical report, NRL Technical Report, 2000.

[25] J. Baker. Generating images for a time multiplexed stereoscopic computer graphics system: True 3D imaging techniques and display technologies. In *Proceedings the International Society for Optical Engineering (SPIE'91)*, 1987.

[26] C. G. Barbour and G. W. Meyer. Visual cues and pictorial limitations for computer generated photo-realistic images. *The Visual Computer: International Journal of Computer Graphics*, 9(3):151–165, 1992.

[27] J. J. Barnette. Effects of stem and Likert response option reversals on survey internal consistency: If you feel the need, there is a better alternative to using those negatively worded stem. *Educational and Psychological Measurement*, 60(3):361–370, 2000.

[28] T. A. Bass. *The Eudaemonic Pie*. Penguin USA, 1st vintage edition, 1992.

[29] M. Bauer, B. Bruegge, G. Klinker, A. MacWilliams, T. Reicher, S. Riss, C. Sandor, and M. Wagner. Design of a component-based augmented reality framework. In *Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR '01)*, 2001.

[30] R. Behringer. Registration for outdoor augmented reality applications using computer vision techniques and hybrid sensors. In *Proceedings of the IEEE Virtual Reality (VR '99)*. IEEE Computer Society, 1999.

[31] S. Bennet, S. McRobb, and R. Farmer. *Object-Oriented Systems Analysis and Design using UML*. McGraw-Hill, 1999.

[32] S. Bennet, J. Skelton, and K. Lunn. *Schaum's outline of UML*. McGraw-Hill, 2nd edition, 2005.

[33] A. Bierbaum, C. Just, P. Harting, K Meinert, A. Baker, and C. Cruz-Neira. VR juggler: a virtual platform for virtual reality application development. In *Proceedings of the Virtual Reality 2001 annual symposium (VR'01)*, 2001.

[34] M. Billinghurst, J. Bowskill, N. Dyer, and J. Morphett. An evaluation of wearable information spaces. In *Proceedings of the Virtual Reality Annual International Symposium (VRAIS '98)*, 1998.

[35] M. Billinghurst, J. Bowskill, M. Jessop, and J. Morphett. A wearable spatial conferencing space. In *Proceedings of the 2nd IEEE International Symposium on Wearable Computers (ISWC '98)*. IEEE Computer Society, 1998.

[36] M. Billinghurst and H. Kato. Asymmetries in collaborative wearable interfaces. In *Proceedings of the 3rd International Symposium on Wearable Computers (ISWC'93)*. IEEE Computer Society, 1999.

[37] M. Billinghurst and H. Kato. Collaborative mixed reality. In *Proceedings of the International Symposium on Mixed Reality (ISMR 99)*, 1999.

[38] B. Boehm and C. Abts. COTS integration: Plug and pray? *Computer*, 32(1):135–138, January 1999.

[39] D. A. Bowman and L. F. Hodges. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *Proceedings of the 1997 symposium on Interactive 3D graphics (SI3D '97)*, 1997.

[40] D. A. Bowman and C. A. Wingrave. Design and evaluation of menu systems for immersive virtual environments. In *Proceedings of the Virtual Reality 2001 Conference (VR'01)*, 2001.

[41] J. Bray and C. Sturman. *Bluetooth: Connect Without Cables*. Prentice Hall, 1st edition, 2000.

[42] G. M. Breakwell, S. Hammond, and C. Fife-Schaw. *Research Methods in Psychology*. Sage Publications, 2nd edition, 2000.

[43] D. E. Breen, R. T. Whitaker, E. Rose, and M. Tuceryan. Interactive occlusion and automatic object placement for augmented reality. *Computer Graphics Forum*, 15(3):11–22, 1996.

[44] J. Brooke, N. Bevan, F. Brigham, S. Harker, and D. Youmans. Usability statements and standardisation: Work in progress in ISO. In *Proceedings of the IFIP TC13 Third Interantional Conference on Human-Computer Interaction (INTERACT '90)*, 1990.

[45] F. P. Brooks. What's real about virtual reality? In *IEEE Computer Graphics and Applications 19,6*, 1999. pp. 16–27.

[46] L. Brownsword, D. Carney, and P. Oberndorf. The opportunities and complexities of applying commercial-off-the-shelf components. *Defence Software Engineering*, 11(4):4–6, April 1998.

[47] L. Brownsword, T. Oberndorf, and C. A. Sledge. Developing new processes for COTS based systems. *IEEE Software*, 17(4):48–55, July/August 2000.

[48] S. Bryson. Effects of lag and frame rate on various tracking tasks. In *Proceedings of the SPIE — Stereoscopic Displays and Applications IV*, 1993.

[49] T. Büren, P. Lukowicz, and G. Troster. Kinetic energy powered computing - an experimental feasibility study. In *Proceedings of the Seventh International Symposium on Wearable Computers (ISWC '03)*. IEEE Computer Society, 2003.

[50] J. Calvin, A. Dickens, B. Gaines, P. Metzger, D. Miller, and D. Owen. The SIMNET virtual world architecture. In *Virtual Reality Annual International Symposium (VRAIS'93)*, 1993.

[51] K. Cheverst, N. Davies, K. Mitchell, A. Friday, and C. Efstratiou. Developing a context-aware electronic tourist guide: Some issues and experiences. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '00)*. IEEE Computer Society, 2000.

[52] J. Child. Wearable computers blaze connectivity path. *COTS Journal*, 5(4), April 2003.

[53] J. H. CLark. Hierarchical geometric models for visible surface algorithms. *Communications of the ACM*, 9(10):547–554, October 1976.

[54] G. M. Clarke and R. E. Kempson. *Design and Analysis of Experiments*. Arnold Publications, 1997.

[55] J. M. Converse and S. Presser. *Survey Questions: Handcrafting the Standardized Questionnaire*. Sage Publications, Paperback edition, 1986.

[56] H. Cooligan. *Introduction to Reseach Methods and Statistics in Psychology*. Hoddler & Stoughton, 2nd edition, 1996.

[57] C. D. Correa and I. Marsic. A simplification architecture for exploring navigation tradeoffs in mobile VR. In *Proceedings of the IEEE Virtual Reality annual symposium (VR'04)*, 2004.

[58] J. W. Cresswell. *Survey Methods in Social Investigation*. Dartmouth, Hardcover edition, 1985.

[59] L. J. Cronbach. Coefficient Alpha and the internal structure of tests. *Psychometrika*, (16):297–334, 1951.

[60] J. Cross. WearCAM personal imaging. Master's thesis, University of Birmingham, 2000.

[61] E. Dallage and G. Vench. Thermal characterization of compact electronic systems: A portable PC as a study case. *IEEE Transactions on Power Electronics*, 17(2):187–195, March 2002.

[62] E. Thorpe Davis and L. F. Hodges. Human stereopsis, fusion, and stereoscopic virtual environments. *Virtual environments and advanced interface design*, pages 145–174, 1995.

[63] D. DeVaus. *Surveys in Social Research*. Routledge, 5th edition, 2004.

[64] R. F. DeVellis. *Scale Development: Theory and Applications*. Sage Publications, Paperback edition, 1991.

[65] A. Dey and G. Abowd. The Context Toolkit aiding the development of context aware applications. In *Proceedings of Human Factors in Computing Systems (CHI '99)*. ACM Press, 1999.

[66] A. Dey, G. Abowd, and D. Salber. A context-based infrastructure for smart environments, 1999. Proceedings of the 1st International Workshop on Managing Interactions in Smart Environments (MANSE '99).

[67] A. K. Dey, D. Salber, G. D. Abowd, and M. Futakawa. The Conference Assistant combining context-awareness with wearable computing. In *Proceedings of the Third International Symposium on Wearable Computers (ISWC '99)*. IEEE Computer Society, 1999.

[68] J. G. Dorsey and D. P. Siewiorek. Online power monitoring for wearable systems. In *Proceedings of the Sixth International Symposium on Wearable Computers (ISWC '02)*. IEEE Computer Society, 2002.

[69] D. Drasic and P. Milgram. Perceptual issues in augmented reality. In *Stereoscopic Displays and Virtual Reality Systems III*, volume 2653, pages 123–134, 1996. Proc. SPIE.

[70] D. G. Elmes, B. H. Kantowitz, and H. L. Roediger III. *Reseach Methods in Psychology*. Brooks/Cole, 6th edition, 1999.

[71] S. Feiner, B. MacIntyre, and T. Höllerer. A Touring Machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment. In *Proceedings of the First International Symposium on Wearable Computers (ISWC '97)*. IEEE Computer Society, 1997.

[72] S. Fickas, G. Kortuem, and Z. Segall. Software organization for dynamic and adaptable wearable systems. In *Proceedings of the 1st IEEE International Symposium on Wearable Computers (ISWC '97)*. IEEE Computer Society, 1997.

[73] A. E. Foni, G. Papagiannakis, and N. Magnenat-Thalmann. Virtual Hagia Sophia: Restitution, visualization and virtual life simulation. In *30th Anniversary Virtual Congress: Architecture, Tourism and World Heritage*. UNESCO, 2002.

[74] L. H. Frank, J. G Casali, and W. W. Wierwille. Effects of visual display and motion system delays on operator performance and uneasiness in a driving simulator. *Human Factors*, 30(2):201–217, April 1988.

[75] E. Frecon and M. Stenius. DIVE: a scaleable network architecture for distributed virtual environments. *Distributed Systems Engineering*, 5(3):91–100, 1998.

[76] H. H. Friedman, P. J. Herskovitz, and S. Pollack. The biasing effects of scale-checking styles on response to a Likert scale. In *Proceedings of the Survey Research Methods Section, ASA*, 1993.

[77] J. Fung, F. Tang, and S. Mann. Mediated reality using computer graphics hardware for computer vision. In *Proceedings of the Sixth International Symposium on Wearable Computers (ISWC '02)*. IEEE Computer Society, 2002.

[78] T. A. Furness. The Super Cockpit and its human factors challenges. In *Proceedings of the 30th Annual Meeting Human Factors Society*. Human Factors Society, 1986.

[79] Y. Genc, F. Sauer, F. Wenzel, M. Tuceryan, and N. Navab. Optical see-through HMD calibration: A stereo method validated with a video see-through system. In *Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR 2000)*, 2000.

[80] T. Gleue and D. Patrick. Design and implementation of a mobile device for outdoor augmented reality in the archeoguide project. In *Proceedings of the 2001 conference on Virtual Reality, Archeology, and Cultural Heritage (VAST '01)*, 2001.

[81] M. Good, T. M. Spine, J. Whiteside, and P. George. User-derived impact analysis as a tool for usability engineering. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '86)*, 1986.

[82] G. Gordon, M. Billinghurst, M. Bell, J. Woodfill, B. Kowalik, A. Erendi, and J. Tillander. The use of dense stereo range data in augmented reality. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR'02)*, 2002.

[83] F. S. Grassia. Practical parameterization of rotations using the exponential map. *Journal of Graphics Tools*, 3(3):29–48, 1998.

[84] B. Grimes. Tech success: 'Heads-up' takes on fresh meaning for the Army. *Washington Technology*, 19(12), 2004.

[85] S. L. Grotch. Three-dimensional and stereoscopic graphics for scientific data display and analysis. *IEEE Computer Graphics Applications*, 3(8):31–43, 1983.

[86] R. Hahn and H. Reichl. Batteries and power supplies for wearable and ubiquitous computing. In *Proceedings of the Third International Symposium on Wearable Computers (ISWC '99)*. IEEE Computer Society, 1999.

[87] J. Hall and R. Naff. The cost of COTS. *IEEE Aerospace and Electronic Systems magazine (AESS)*, 16(8):20–24, August 2001.

[88] L. Hasig, editor. *Air Combat*. The New Face of War. Time-Life Books, 1990.

[89] L. Hasig, editor. *The Armored Fist*. The New Face of War. Time-Life Books, 1990.

[90] J. Healey and R.W. Pickard. A Cybernetic wearable camera. In *Proceedings of the Second International Symposium on Wearable Computers (ISWC '98)*. IEEE Computer Society, 1998.

[91] M. L. Heilig. United States Patent office: stereoscopic-television apparatus for individual use. *ACM SIGGRAPH Computer Graphics*, 28(2):131–134, 1994.

[92] R. Held and N. Durlach. *Pictorial communication in virtual and real environments*. Taylor and Francis, Inc., 2nd edition, 1993.

[93] D. Henry and T. Furness. Spatial perception in virtual environments: Evaluating and architectural application. In *Proceedings of the IEEE Virtual Reality annual symposium (VR'93)*, 1993.

[94] G. Hesina, D. Schmalstieg, A. Furhmann, and W. Purgathofer. Distributed Open Inventor: a practical approach to distributed 3D graphics. In *Proceedings of the ACM symposium on Virtual Reality Software and Technology (VRST '99)*, 1999.

[95] L. F. Hodges. Tutorial: Time-multiplexed stereoscopic computer graphics. *IEEE Computer Graphics Applications*, 12(2):20–30, 1992.

[96] L. F. Hodges and D. F. McAllister. Computing stereographic views. In *Stereographics: ACM SIGGRAPH '89 Course Notes*, 1989.

[97] L. F. Hodges and D. F. McAllister. Rotation algorithm artifacts in stereoscopic images. *Optical Engineering*, 29(8):973–976, August 1990.

[98] T. Höllerer, S. Feiner, T. Terauchi, G. Rashid, and D. Hallaway. Exploring MARS: Developing indoor and outdoor user interfaces to a mobile augmented reality system. *Computers and Graphics*, 23(6):779–785, 1999.

[99] T. Höllerer, D. Hallaway, N. Tinna, and S. Feiner. Steps towards accommodation variable position tracking accuracy in a mobile augmented reality system. In *Proceedings of the second international workshop on Artificial Intelligence in Mobile Systems (AIMS'01)*, 2001.

[100] T. Höllerer and J. Pavlik. Situated Documentaries: Embedding multimedia presentations in the real world. In *Proceedings of the Third International Symposium on Wearable Computers (ISWC '99)*. IEEE Computer Society, 1999.

[101] H. Holloway. Registration error analysis for augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):413–432, 1997.

[102] I. P. Howard and B. J. Rogers. *Binocular Vision and Stereopsis*. Oxford University Press, 1995.

[103] R. Hubbold, J. Cook, M. Keates, S. Gibson, T. Howard, A. Murta, A. West, and S. Pettifer. GNU/MAVERIK: a micro-kernel for large-scale virtual environments. In *Proceedings of the ACM symposium on Virtual Reality software and technology (VRST '99)*, 1999.

[104] R. Hubbold, D. Hancock, and C. Moore. Stereoscopic volume rendering. In *Proceedings of the Visualization in Scientific Computing '98*. Springer-Verlag, 1998.

[105] R. Hubbold, A. Murta, A. West, and T. Howard. Design issues for virtual reality systems. In *Selected papers of the Eurographics workshops on Virtual environments '95 (VE '95)*. Springer-Verlag, 1995.

[106] IEEE. *830-1998 Recommended Practice for Software Requirements Specifications (ANSI/IEEE)*. IEEE, October 1998.

[107] H. Irie and K. Imamura. DGPS for outdoor mobile robots by a wireless lan. *Journal of Natural Sciences*, 8(2B):581–586, June 2003.

[108] ISO. *International Organization for Standardization. ISO 15469:2004. Spatial distribution of daylight – CIE standard general sky*. International Organization for Standardization, 1997.

[109] ISO. *International Organization for Standardization. ISO9241-3. Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs). Part 3 Visual Display Requirements*. International Organization for Standardization, 1997.

[110] S. Jamieson. Likert scales: how to (ab)use them. *Medical Education*, 38(12):1217–1218, 2004.

[111] D. J. Johnston. *Position sensing and augmented reality*. PhD thesis, University of Essex, 2004.

[112] R. S. Kalawsky. *The Science of Virtual Reality and Virtual Environments*. Addison-Wesley, 1st edition, 1993.

[113] R. S. Kalawsky. New methodologies and techniques for evaluating user performance in advanced 3D virtual interfaces. IEE Colloquium on the 3D Interface for the Information Worker (Digest No. 1998/437), May 1998.

[114] R. S. Kalawsky. VRUSE—a computerised diagnostic tool: For usability evaluation of virtual/synthetic environment systems. *Special Edition of Applied Ergonomics*, 30(1):11–25, January 1999.

[115] R. S. Kalawsky, S. T. Bee, and S. P. Nee. Human factors evaluation techniques to aid understanding of virtual interfaces. *BT Technology Journal*, 17(1):128–141, 1999.

[116] R. Kalden, I. Meirick, and M. Meyer. Wireless internet access based on GPRS. *IEEE Personal Communications*, 7(2):8–1–8, April 2000.

[117] H. Kato and M. Billinghurst. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR '99)*. IEEE Computer Society, 1999.

[118] K. Keller and D. Colucci. Perception in HMDs what is it in head mounted displays that really make them all so terrible? Helmet and Head-Mounted Displays III, April 1998. pages 46–53.

[119] J. Kelso, S. G. Satterfield, L. E. Arsenault, P. M. Ketchan, and R. D. Kriz. DIVERSE: a framework for building extensible and reconfigurable device-independent virtual environments and distributed asynchronous simulations. *Presence: Teleoperators and Virtual Environonments.*, 12(1):19–36, 2003.

[120] R. S. Kennedy, N. E. Lane, K. S. Berbaum, and M. G. Lilienthal. Simulator sickness questionnaire: An enhanced method for quantifying simulator sickness. *The International Journal of Aviation Psychology*, 3(3):203–222, 1993.

[121] R. S. Kennedy, N. E. Lane, M. G. Lilienthal, K. S. Berbaum, and L. J. Hettinger. Profile analysis of simulator sickness symptoms: application to virtual environment systems. *Presence: Teleoperators and Virtual Environments*, 1(3):295–301, 1992.

[122] B. W. Knerr, P. J. Garrity, and D. R. Lampton. Embedded training for Future Force Warriors: An assessment of wearable virtual simulators. In *24th Army Science Conference Proceedings*, 2004.

[123] J. F. Knight and C. Baber. Neck muscle activity and perceived pain and discomfort due to variations of head load and posture. *Aviation, Space, and Environmental Medicine,*, 75(2):123–131, February 2004.

[124] J. F. Knight, C. Baber, A. Schwirtz, and H.W. Bristow. The comfort assessment of wearable computers. In *Proceedings of the Sixth International Symposium on Wearable Computers (ISWC '02)*, 2002.

[125] J. Kollin. A retinal display for virtual-environment applications. In *Proceedings of the International Symposium of the Society for Information Display*, 1993.

[126] G. Kortuem, M. Bauer, and Z. Segall. NETMAN: the design of a collaborative wearable computer system. *Mobile Network Applications*, 4(1):49–58, 1999.

[127] K. N. Kutulakos and J. Vallino. Affine object representations for calibration-free augmented reality. In *Proceedings of the 1996 Virtual Reality Annual International Symposium (VRAIS 96)*. IEEE Computer Society, 1996.

[128] K. N. Kutulakos and J. R. Vallino. Calibration-free augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, 4(1):1–20, 1998.

[129] J. Kymissis, Clyde Kendall, Joseph Paradiso, and Neil Gershenfeld. Parasitic power harvesting in shoes. In *Proceedings of the Second International Symposium on Wearable Computers (ISWC '98)*. IEEE Computer Society, 1998.

[130] K. Van Laerhoven, K. A.Aidoom, and S. Lowette. Real-time analysis of data from many sensors with neural networks. In *Proceedings of the Fifth International Symposium on Wearable Computers (ISWC '01)*. IEEE Computer Society, 2001.

[131] S. Lee and K. Mase. Incremental motion based location recognition. In *Proceedings of the Fifth International Symposium on Wearable Computers (ISWC '01)*. IEEE Computer Society, 2001.

[132] V. Lepetit and M.O. Berger. Handling occlusion in augmented reality systems: A semi-automatic method. In *IEEE and ACM International Symposium on Augmented Reality (ISAR'00)*, page 137. IEEE Computer Society, 2000.

[133] J. Liang, C. Shaw, and M. Green. On temporal-spatial realism in the virtual reality environment. In *Proceedings of the 4th annual ACM symposium on User interface software and technology (UIST '91)*, 1991.

[134] M. A. Livingston, L. J. Rosenblum, S. J. Julier, Y. Baillot D. Brown, J. E. Swan Ii, J. L. Gabbard, and D. Hix. An augmented reality system for military operations in urban terrain. In *Proceedings of the Interservice, Industry Training, Simulation and Education Conference (IITSEC'02)*. IEEE Computer Society, 2002.

[135] K. Lyons and T. Starner. Mobile capture for wearable computer usability testing. In *Proceedings of the 5th IEEE International Symposium on Wearable Computersn (ISWC '01)*, 2001.

[136] M. R. Macedonia, M. J. Zyda, D. R. Pratt, P. T. Barham, and S. Zeswitz. NPSNET: A network software architecture for large-scale virtual environment. *Presence: Teleoperators and Virtual Environments*, 3(4):265–287, 1994.

[137] B. MacIntyre and S. Feiner. Language-level support for exploratory programming of distributed virtual environments. In *Proceedings of the 9th annual ACM symposium on User Interface Software and Technology (UIST '96)*, pages 83–94, 1996.

[138] I. S. MacKenzie and C. Ware. Lag is a determinant of human performance in interactive systems. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '93)*, 1993.

[139] S. Malik, C. McDonald, and G. Roth. Hand tracking for interactive pattern-based augmented reality. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR 2002)*. IEEE Computer Society, 2002.

[140] S. Mann. An historical account of 'Wearcomp' and 'Wearcam' inventions developed for applications in 'Personal Imaging"'. In *Proceedings of the First International Symposium on Wearable Computers (ISWC '97)*. IEEE Computer Society, 1997.

[141] S. Mann. Smart clothing: The wearable computer and wearcam. *Personal Technologies*, 1(1), March 1997.

[142] S. Mann. Definition of wearable computer. In *Proceedings of the Second International Symposium on Wearable Computers (ISWC '98)*. IEEE Computer Society, 1998.

[143] S. Mann. *Intelligent image processing*. Wiley, New York, 2002.

[144] T. L. Martin and D. P. Siewiorek. Non-ideal battery properties and low power operation in wearable computing. In *Proceedings of the Third International Symposium on Wearable Computers (ISWC '99)*. IEEE Computer Society, 1999.

[145] G. Mather and D.R.R Smith. Depth cue integration: stereopsis and image blur. *Vision Research*, 40(25):3501–3506, 2000.

[146] W. D. McCarty, S Sheasby, P. Amburn, M.R. Stytz, and C. Switzer. A virtual cockpit for a distributed interactive simulation. *IEEE Computer Graphics and Applications*, 14(1):49–59, January 1994.

[147] J. McHale. Wired and ready to wear. *Military and Aerospace Electronics*, November 2004.

[148] J. Melzer and K. Moffitt. *Head-Mounted Displays: Designing for the User*. McGraw-Hill, 1st edition, 1996.

[149] K. Meyer, H.L. Applewhite, and F.A. Biocca. A survey of position trackers. In *Presence: Teleoperators and Virtual Environments, 1.2,* pages pp. 173–200. MIT Press, 1992.

[150] P. Milgram and F. Kishino. A taxonomy of mixed reality visual displays. *IEICE Trans. Information Systems, vol. E77-d, no.12*, 1994. pp. 1321–1329.

[151] M. H. Morgan. *Vitruvius, the Ten Books on Architecture*. Dover publications, 1967.

[152] Sir. C. Moser and G. Kalton. *Survey Methods in Social Investigation*. Dartmouth, Hardcover edition, 1985.

[153] C. E. Moxey. *Periscopic stereo and large-scene reconstruction*. PhD thesis, University of Essex, 2002.

[154] J. Murray. Wearable computers in battle: Recent advances in the Land Warrior system. In *Proceedings of the Fourth International Symposium on Wearable Computers (ISWC '00)*. IEEE Computer Society, 2000.

[155] H. Nagahara, Y. Yagi, and M. Yachida. Wide field of view head mounted display for telepresence with an omnidirectional image sensor. In *Conference on Computer Vision and Pattern Recognition Workshop*, volume 7, page 86, 2003.

[156] J. Newman, D. Ingram, and A. Hopper. Augmented reality in wide area sentient environment. In *Proceedings of the 2nd IEEE and ACM International Symposium on Augmented Reality (ISAR'01)*. IEEE Computer Society, 2001.

[157] N. J. Newman. *Systems and Services for Wearable Computers*. PhD thesis, University of Essex, 2002.

[158] N. J. Newman and A. F. Clark. Sulawesi: A wearable application integration framework. In *Proceedings of the Third International Symposium In Wearable Computers (ISWC '99)*. IEEE Computer Society, 1999.

[159] R. Pausch, T. Crea, and M. Conway. A literature survey for virtual environments: military flight simulator visual systems and simulator sickness. *Presence: Teleoperators and Virtual Environments*, 1(3):344–363, 1992.

[160] R. Pausch, M.A. Shackelford, and D. Proffitt. A user study comparing head-mounted and stationary displays. In *IEEE 1993 Symposium on Research Frontiers in Virtual Reality*, 1993.

[161] PC/104 Embedded Consortium. *PC/104-Plus specification*, November 2003. Version 2.5.

[162] PC/104 Embedded Consortium. *PC/104 specification*, November 2003. Version 2.0.

[163] H. T. Peck. *Harper's Dictionary of Classical Literature and Antiquities*. Cooper Square Publishers, Inc., 2nd edition, 1962.

[164] R. W. Picard and J. Healey. Affective wearables. In *Proceedings of the First International Symposium on Wearable Computers (ISWC '97)*. IEEE Computer Society, 1997.

[165] W. Piekarski. *Interactive 3D Modelling in Outdoor Augmented Reality Worlds*. Phd thesis, University of South Australia, 2004.

[166] W. Piekarski, B. Avery, B.H. Thomas, and P. Malbezin. Hybrid indoor and outdoor tracking for mobile 3D mixed reality. In *Proceedings of the 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR2003)*, October 2003.

[167] W. Piekarski and B. H. Thomas. Tinmith Metro: New outdoor techniques for creating city models with an augmented reality wearable computer. In *Proceedings of the Fifth International Symposium on Wearable Computers (ISWC '01)*. IEEE Computer Society, 2001.

[168] W. Piekarski and B.H. Thomas. Interactive augmented reality techniques for construction at a distance of 3D geometry. In *Proceedings of the workshop on Virtual environments 2003 (EGVE '03)*. ACM Press, 2003.

[169] W. Piekarski and B.H. Thomas. An object-oriented software architecture for 3D mixed reality applications. In *Proceedings of the The 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR '03)*. IEEE Computer Society, 2003.

[170] W. Piekarski and B.H. Thomas. Tinmith — A mobile outdoor augmented reality modelling system. In *Symposium on Interactive 3D Graphics*, 2003.

[171] W. Piekarski and B.H. Thomas. Tinmith — Mobile outdoor augmented reality modelling demonstration. In *Proceedings of the 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR '03)*, October 2003.

[172] W. Piekarski, B.H. Thomas, D. Hepworth, G. Gunther, and V. Demczuk. An architecture for outdoor wearable computers to support augmented reality and multimedia applications. In *Proceedings of the 3rd international conference on Knowledge-Based Intelligent Information Engineering Systems (KES'99)*, 1999.

[173] W. Piekarsky, B. Avery, H.H. Thomas, and P. Malbezin. Integrated head and hand tracking for indoor and outdoor augmented reality. In *Proceedings of the IEEE Virtual Reality conference (VR'04)*, 2004.

[174] N. Pollard, J. K. Hodgins, M.J. Riley, and C. G. Atkeson. Adapting human motion for the control of a humanoid robot. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '02)*, May 2002.

[175] J. Preece, Y. Rogers, H. Sharp, D. Benyon, S. Holland, and T. Carey. *Human-Computer Interaction*. Addison-Wesley, 1st edition, April 1994.

[176] S. Presser and H. Schuman. The measurement of the middle position. *Public Opinion Quarterly*, (44):70–85, 1980.

[177] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket location-support system. In *Proceedings of the 6th annual international conference on Mobile computing and networking (MobiCom '00)*, 2000.

[178] J. Prothero, H.G. Hoffman, D.E. Parker, M. Wells, and T.A. Furness. Foreground/background manipulations affect presence. In *In Proceedings of Human Factors and Ergonomics Society (HFES'95)*, pages 1410–1414, 1995.

[179] H. Neely III R. Azuma, B. Hoff and R. Sarfaty. A motion-stabilized outdoor augmented reality system. In *Proceedings of the IEEE Virtual Reality annual symposium (VR'99)*, 1999.

[180] P. Rademacher, J. Lengyel, E. Cutrell, and T. Whitted. Measuring the perception of visual realism in images. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, 2001.

[181] J. J. Ray. Acquiescence and problems with forced-choice scales. *Journal of Social Psychology*, 130(3):397–399, 1990.

[182] M. Reddy. The effects of low frame rate on a measure for user performance in virtual environments. Technical Report ECS-CSG-36-97, University of Edinburgh, December 1997.

[183] M. Reddy. *Perceptually Modulated Level of Detail for Virtual Environments*. Phd thesis, University of Edinburgh, 1997.

[184] C. Regan. An investigation into nausea and other side-effects of head-coupled immersive virtual reality. Virtual Reality: Research, Development, Applications, 1995.

[185] G. Reitmayr and D. Schmalstieg. OpenTracker—an open software architecture for reconfigurable tracking based on XML. In *Proceedings of the Virtual Reality 2001 annual symposium (VR'01)*, pages 285–286, 2001.

[186] G. Reitmayr and D. Schmalstieg. A wearable 3D augmented reality workspace. In *Proceedings of the 5th IEEE International Symposium on Wearable Computers (ISWC '01)*, 2001.

[187] B. J. Rhodes. The wearable remembrance agent: a system for augmented memory. In *Proceedings of the First International Symposium on Wearable Computers (ISWC '97)*. IEEE Computer Society, 1997.

[188] B. J. Rhodes, N. Minar, and J. Weaver. Wearable computing meets ubiquitous computing: Reaping the best of both worlds. In *Proceedings of the Third International Symposium on Wearable Computers (ISWC '99)*. IEEE Computer Society, 1999.

[189] B.J. Rhodes. WIMP interface considered fatal. *In IEEE Virtual Reality Annual International Symposium (VRAIS'98) — Workshop on Interfaces for Wearable Computers*, 1998.

[190] M. Ribo, P. Lang, H. Ganster, M. Brandner, C. Stock, and A. Pinz. Hybrid tracking for outdoor augmented reality applications. *IEEE Computer Graphpics and Applications*, 22(6):54–63, 2002.

[191] P. Richard, G. Burdea, G. Birebent, D. Gomez, N. Langrana, and P. Coiffet. Effect of frame rate and force feedback on virtual object manipulation. *Presence: Teleoperators and Virtual Environments*, 5(1):95–108, 1996.

[192] P. Ritsos, D. J. Johnston, C. Clark, and A. F. Clark. Mobile augmented reality archaeological reconstruction. In *30th Anniversary Virtual Congress: Architecture, Tourism and World Heritage*. UNESCO, 2002.

[193] P. Ritsos, D. J. Johnston, C. Clark, and A. F. Clark. Engineering an augmented reality tour guide. In *Eurowearable '03*. IEEE Computer Society, 2003.

[194] J. Rohlf and J. Helman. IRIS Performer: A high performance multiprocessing toolkit for real-time 3D graphics. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques (SIGGRAPH '94)*, 1994.

[195] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. Szalavri, L. M. Encarnao, M. Gervautz, and W. Purgathofer. The Studierstube augmented reality project. *Presence: Teleoperators and Virtual Environments*, 11(1):33–54, 2002.

[196] P. Scholfield. *Quantifying Language: A Researcher's and Teacher's Guide to Gathering Language Data and Reducing It to Figures*. Sage Publications, 2nd edition, 2000.

[197] S. J. Schwartz and A. Pentland. The Smart Vest: Towards a next generation wearable computing platform. Technical Report 504, MIT Media Laboratory Perceptual Computing Section, July 1999.

[198] V. Schwieger. Using handheld GPS receivers for precise positioning. In *Proceedings of the 2nd FIG Regional Conference in Marrakech*, 2003.

[199] C. Seale. *Researching Society and Culture*. Sage Publications, Hardcover edition, 2004.

[200] J. J. Shaughnessy, E.B. Zechmeister, and J.S. Zechmeister. *Reseach Methods in Psychology*. McGraw-Hill, 5th edition, 2000.

[201] C. Shaw and M. Green. The MR toolkit Peers Package and experiment. In *Proceedings of Virtual Reality Annual International Symposium (VRAIS '93)*, 1993.

[202] T. B. Sheridan. Musings on telepresence and virtual environments. *Presence: Teleoperators and Virtual Environments*, 1(1):120–126, 1992.

[203] D. Shreiner, M. Woo, J. Neider, and T. Davis. *OpenGL Programming Guide: The Official Guide to Learning OpenGL*. Addison-Wesley, 4th edition, 2003.

[204] J. Van Sickle. *GPS for Land Surveyors*. CRC Press, 2nd edition, 2001.

[205] J. Siegel and M. Bauer. A field usability evaluation of a wearable system. In *Proceedings of the 1st IEEE International Symposium on Wearable Computers (ISWC '97)*, 1997.

[206] S. Siegel and N. J. Castellan. *Nonparametric statistics for the Behavioral Sciences*. McGraw-Hill, Paperback edition, March 1988.

[207] S. K. Singhal and D. R. Cheriton. Exploiting position history for efficient remote rendering in networked virtual reality. *Presence: Teleoperators and Virtual Environments*, 4(2):169–193, 1995.

[208] M. Slater. Measuring Presence: A response to the Witmer and Singer Presence questionnaire. *Presence*, 8(5):560–565, 1999.

[209] M. B. Spitzer, N.M. Rensing, and R. McClelland. Eyeglass-based systems for wearable computing. In *Proceedings of the First International Symposium on Wearable Computers (ISWC '97)*. IEEE Computer Society, 1997.

[210] T. Staner, D. Kirsch, and S. Assefa. The Locust swarm: An environmentally-powered, networkless location and messaging system. In *Proceedings of the First International Symposium on Wearable Computers (ISWC '97)*. IEEE Computer Society, 1997.

[211] T. Starner. Wearable computing and augmented reality. Technical Report 355, MIT Media Lab Vision and Modeling Group, 1995.

[212] T. Starner. The challenges of wearable computing: Part 1. *IEEE Micro*, 21(4):44–52, July 2001.

[213] T. Starner. The challenges of wearable computing: Part 2. *IEEE Micro*, 21(4):54–67, July 2001.

[214] P. S. Strauss. IRIS Inventor, a 3D graphics toolkit. In *Proceedings of the 8th Annual Conference on Object-oriented programming systems, languages, and applications (OOPSLA '93)*, 1993.

[215] S. Sudman and N. M. Bradburn. *Asking Questions : A Practical Guide to Questionnaire Design*. Jossey-Bass, 1st edition, 1982.

[216] R. Suomela and J. Lehikoinen. Context compass. In *Proceedings of the Fourth International Symposium on Wearable Computers (ISWC '00)*. IEEE Computer Society, 2000.

[217] R. Suomela, J. Lehikoinen, and I. Salminen. A system for evaluating augmented reality user interfaces in wearable computers. In *Proceedings of the Fifth International Symposium on Wearable Computers (ISWC '01)*. IEEE Computer Society, 2001.

[218] I. E. Sutherland. The Ultimate Display. In *Proceedings of IFIP Congress*, 1965.

[219] A. Tang, J. Zhou, and C. Owen. Evaluation of calibration procedures for optical see-through head-mounted displays. In *Proceedings of the The 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR '03)*. IEEE Computer Society, 2003.

[220] R. M. Taylor, T. C. Hudson, A. Seeger, H. Weber, J. Juliano, and A. T. Helser. VRPN: a device-independent, network-transparent vr peripheral system. In *Proceedings of the ACM symposium on Virtual Reality software and technology (VRST '01)*, 2001.

[221] S. Thad, S. Mann, B. Rhodes, J. Levine, J. Healey, D. Kirsch, R. Picard, and A. Pentland. Augmented reality through wearable computing. *Presence: Teleoperators and Virtual Environments*, 6(4):386–398, August 1997.

[222] B. Thomas, B. Close, J. Donoghue, J. Squires, P. de Bondi, M. Morris, and W. Piekarski. ARQuake: An outdoor/indoor augmented reality first person application. In *Proceedings of the 4th IEEE International Symposium on Wearable Computers (ISWC '00)*. IEEE Computer Society, 2000.

[223] B. Thomas, V. Demeczuk, W. Piekarski, D. Hepworth, and B. Gunther. A wearable computer system with augmented reality to support terrestrial navigation. In *Proceedings of the Second International Symposium on Wearable Computers (ISWC '98)*. IEEE Computer Society, 1998.

[224] B. Thomas, S. Tyerman, and K. Grimmer. Evaluation of three input mechanisms for wearable computers. In *Proceedings of the First International Symposium on Wearable Computers (ISWC '97)*. IEEE Computer Society, 1997.

[225] E. O. Thorp. *Beat the Dealer*. Random House, Paperback edition, 1962.

[226] E. O. Thorp. The invention of the first wearable computer. In *Proceedings of the Second International Symposium on Wearable Computers (ISWC '98)*. IEEE Computer Society, 1998.

[227] M. Tidwell, R. S. Johnston, D. Melville, and T. A. Furness. The virtual retinal display — a retinal scanning imaging system. In *Virtual Reality World '95 Conference Documentation, IDG Conferences*, 1995.

[228] J. Torborg and J. T. Kajiya. Talisman: commodity realtime 3D graphics for the PC. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques (SIGGRAPH '96)*, pages 353–363, 1996.

[229] H. Tramberend. Avocado: A distributed virtual reality framework. In *Proceedings of the IEEE Virtual Reality annual symposium (VR '99)*, 1999.

[230] P. R. Tregenza. Measured and calculated frequency distributions of daylight illuminance. *Lighting and Research Technology*, 18(2):71–74, 1986.

[231] M. Tuceryan and N. Navab. Single point active alignment method (SPAAM) for optical see-through HMD calibration for AR. In *Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR 2000)*, 2000.

[232] D. A. Tyldesley. Employing usability engineering in the development of office products. *The Computer Journal*, 31(5):431–436, 1988.

[233] A. Valejo, J. Lobo, and J. Dias. Short-range DGPS for mobile robots with wireless Ethernet links. In *5th International Workshop on Advanced Motion Control AMC '98-Coimbra*, pages 334–339, July 1998.

[234] R. De Vaul, M. Sung, J. Gips, and A. Petland. MITHrill 2003: Applications and architecture. In *Proceedings of the Seventh International Symposium on Wearable Computers (ISWC '03)*. IEEE Computer Society, 2003.

[235] V. Vlahakis, J. Karigiannis, M. Tsotros, N. Ioannidis, and D. Stricker. Personalized augmented reality touring of archaeological sites with wearable and mobile computers. In *Proceedings of the Sixth International Symposium on Wearable Computers (ISWC '02)*. IEEE Computer Society, 2002.

[236] D. Wagner and D. Schmalstieg. First steps towards handheld augmented reality. In *Proceedings of the Seventh International Symposium on Wearable Computers (ISWC '03)*. IEEE Computer Society, 2003.

[237] J. Wann, S. Rushton, and M. Mon-Williams. Natural problems for stereoscopic depth perception in virtual environments. *Vision Research*, 35:2731–2736, 1995.

[238] R. Y Want, A. Hopper, V. Falco, and J. Gibbons. The active badge location system. *ACM Transactions on Information Systems*, 10(1):91–102, 1992.

[239] C. Ware, K. Arthur, and K. S. Booth. Fish tank virtual reality. In *Proceedings of the SIGCHI conference on Human factors in computing system (CHI '93)*, pages 37–42, 1993.

[240] C. Ware and R. Balakrishnan. Reaching for objects in VR displays: lag and frame rate. In *ACM Transactions on Computer-Human Interaction (TOCHI)*, 1994.

[241] K. Watsen and M. Zyda. Bamboo — a portable system for dynamically extensible, real-time, networked, virtual environments. In *Proceedings of the Virtual Reality annual international symposium (VRAIS '98)*, 1998.

[242] B. Watson, V. Spaulding, N. Walker, and W. Ribarsky. Evaluation of the effects of frame time variation on VR task performance. In *Proceedings of the 1997 Virtual Reality Annual International Symposium (VRAIS '97)*. IEEE Computer Society, 1997.

[243] G. Welch and G. Bishop. An introduction to the Kalman filter. Technical Report TR 95-041, University of North Carolina - Department of Computer Science, 1995.

[244] R. B. Welch. *Perceptual Modification: Adapting to Altered Sensory Environments*. Elsevier Science and Technology Books, 1978.

[245] R. Westermann and T. Ertl. Efficiently using graphics hardware in volume rendering applications. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques (SIGGRAPH '98)*, pages 169–177, 1998.

[246] J. Willmott, L.I. Wright, D.B.Arnold, and A.M.Day. Rendering of large and complex urban environments for real time heritage reconstructions. In *Proceedings of the IEEE Virtual Reality annual symposium (VR'01)*, 2001.

[247] A. Wilson, M. C. Lin, B.L. Yeo, M. Yeung, and D. Manocha. A video-based rendering acceleration algorithm for interactive walkthroughs. In *Proceedings of the eighth ACM international conference on Multimedia (MULTIMEDIA '00)*. ACM Press, 2000.

[248] B.G. Witmer and M.J. Singer. Measuring presence in virtual environments: a presence questionnaire. *Presence: Teleoperators and Virtual Environments*, 7(3):225–240, June 1998.

[249] M. Wloka. Lag in multiprocessor virtual reality. *Presence: Teleoperators and Virtual Environments, 4.1*, 4:50–63, 1995.

[250] R. L. Woods, I. Fetchenheuer, F. Vargas-Martin, and E. Peli. The impact of non-immersive head-mounted displays (HMD) on the visual field. *Special issue on the International Symposium of the Journal of the Society for Information Display*, 2002.

[251] R. L. Woods, I. Fetchenheuer, F. Vargas-Martin, and E. Peli. The impact of non-immersive head-mounted displays on the visual field. *Journal of The Society for Information Display*, 11(1):191–198, 2003.

[252] M. Wöpking. Viewing comfort with stereoscopic pictures: An experimental study on the subjective effects of disparity magnitude and depth focus. *Journal of The Society for Information Display*, 3(3):101–103, 1995.

[253] J. Yang, W. Yang, and M. Debecke. Smart Sight: A tourist assistant system. In *Proceedings of the Third International Symposium on Wearable Computers (ISWC '99)*. IEEE Computer Society, 1999.

[254] S. You, U. Neuman, and R. Azuma. Hybrid inertial and vision tracking for augmented reality registration. In *Proceedings of the IEEE Virtual Reality annual symposium (VR'99)*, pages 260–, 1999.

[255] S. You, U. Neuman, and R. Azuma. Orientation tracking for outdoor augmented reality registration. In *IEEE Computer Graphics and Applications*, pages 36–42, 1999.

[256] X. Zhong, P. Boulanger, and N.D. Georganas. A collaborative augmented reality system with wearable computing. In *Proceedings of the 21st Biennal Symposium on Communications*, 2002.

[257] M. J. Zieniewicz, D. C. Johnson, C. Wong, and J. D. Flatt. The evolution of army wearable computers. *Pervasive Computing IEEE*, 1(4):30–40, October 2002.