An Architecture for Outdoor Wearable Computers to Support Augmented Reality and Multimedia Applications

Wayne Piekarski, Bruce Thomas, David Hepworth, Bernard Gunther, Victor Demczuk
School of Computer and Information Science – Wearable Computer Lab
University of South Australia, The Levels, SA, Australia
wayne.piekarski@cs.unisa.edu.au

Keywords: Augmented Reality, Wearable Computers, Outdoor Navigation

Abstract

This paper describes an architecture to support a hardware and software platform for research into the use of wearable computers and augmented reality in an outdoor environment. The architecture supports such devices as a GPS, compass, and head-mounted display. A prototype system was built to support novel applications by drawing graphics (such as maps, building plans, and compass bearings) on the head-mounted display, projecting information over that normally seen by the user and hence augmenting a user's perception of reality. This paper presents a set of novel augmented reality and multimedia applications operated in an outdoor environment.

1 Introduction

In recent times, the availability of small and power efficient PC components has opened up new application possibilities for computers. Substantial computing power is no longer confined to the desktop; a new computing paradigm where computers are placed on the user's body, known as *wearable computing*, is now possible. [MANN96, BASS97]

Couple this with recent advances in GPS, head mounted display (HMD), and digital compass technology, it is possible to develop an augmented reality [AZUM97] system suitable to be operated in an outdoor environment. [FEIN97]

Augmented reality is the process of a user viewing the physical world and virtual information simultaneously, where the virtual information is overlaid and aligned to the physical world.

Currently, at the Wearable Computer Laboratory at the University of South Australia, we are conducting research into the hardware and software needed for such a system. [PIEK98b, THOM97, THOM98]

In 1998, we designed and constructed a complete research system, Tinmith-1 [PIEK98a]. This paper defines the overall system architecture we developed to support multimedia and augmented reality applications on a wearable computer platform. The first application domain was a navigation system for a user to wear *outdoors*, which would provide a head up display, similar to that found in an aircraft cockpit, directing the user to known fixed locations. Figure 1 pictures the current hardware platform.

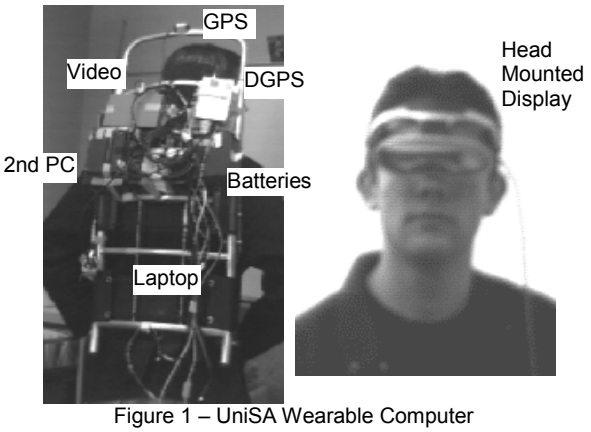

Figure 1 – UniSA Wearable Computer

This paper begins with a description of our hardware platform, followed by the user interface displays, and then has a description of the system architecture, along with some of its innovative features. These allow us to build a multimedia system that combines video, sound, and the real world for the user.

2 Hardware Components

The main- component of our wearable computer system is a Toshiba 320CDS notebook running the freely available LinuxOS. The laptop is about the size of an A4 book and fits comfortably on the wearer's back. The key to wearable computers is its hands free nature, and as a result, we have added various components to allow the user to interact with the system.

Instead of a monitor, we have attached a colour Sony PLM-100 transparent display, which allows the video output of the laptop to superimpose images over the real world. The user wears this display on the head, as shown in Figure 1. Also attached to the computer is a miniature keyboard that attaches to the forearm, which allows the user to interact with the system and enter commands.

By itself, a wearable computer with display and keyboard is useful, but we wanted to use the hardware for something that truly took advantage of the mobile equipment, rather than using it for conventional desktop tasks. Consequently, the first application developed is a navigation system that would present cues onto the display, and guide the user on journeys through places without physical

landmarks or perhaps in darkness. Working with the Defence Science Technology Organisation (DSTO) in South Australia, this is an application for soldiers patrolling in the outdoors. To support the navigation system, a GPS module (with differential receiver) is attached to the laptop, providing position fixes at many places in the world to within 5 metres accuracy. A 3-axis digital compass, also attached to the display, allows the computer to determine exactly how the wearer's head is oriented relative to the surface of the Earth. This information is used to draw the display to match the physical world the user is viewing.

The equipment was attached to a rigid backpack, along with batteries and antennae. The prototype hardware and software system is fully functional in an outdoor environment, and has been taken outside for field trials around the campus, and its environs.

3 Head-Up Displays and Interfaces

The system contains a number of different interfaces to present to the user, and these are covered in the following subsections.

3.1 2D Displays

The 2D interface incorporates a first person perspective, gods-eye view, and traditional non-spatially aware information on one display, shown as Figure 2.

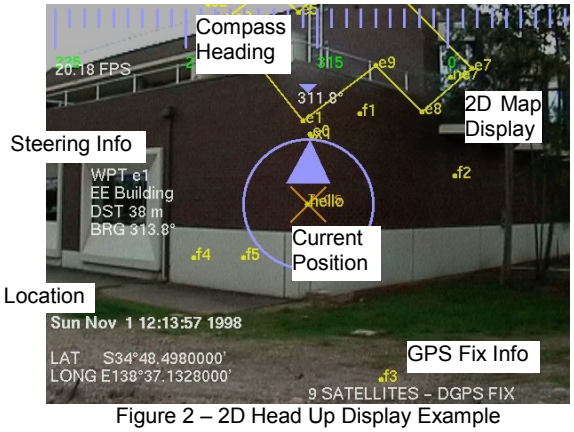

Figure 2 – 2D Head Up Display Example

At the top of the display is the compass heading, which is represented as notches and a value every 45 degrees. As the user rotates their head, the compass updates itself and scrolls left or right to indicate the new heading. The various pieces of text information placed around the display are used to show position, GPS accuracy, and steering instructions to the nominated waypoint.

Underneath the text is the map and navigation gadget display. At the centre of the display is an X, indicating the current position of the system and user. Shown in the figure is the outline of a building, and the circular object in the centre of the screen gives steering instructions to the user as to which direction they must turn to reach the target. The entire display is presented as a gods-eye 2D top down view, where the direction the wearer is facing is up. Every visual cue is rotated in real time as the user moves around.

This display was used for several experiments outdoors. In one case, we entered the dimensions of buildings and walked around them with a hood over our head to test how accurate the system was. In most trials, we maintained better than about 2 metre accuracy assuming we had a good fix with 6 or more GPS satellites.

The display shown in Figure 2 is only one example of the views possible with the system. The software that renders the display is actually broken down into a number of components, which can be readily plugged in and out according to what the user wants. For example, the circular steering cue in the middle can be replaced with an arrow pointing left and right, a bar, or a moving diamond.

3.2 3D Display

We are currently experimenting with a 3D immersive display, where the text and lines drawn on the display actually outline and overlay what the user sees in front of them.

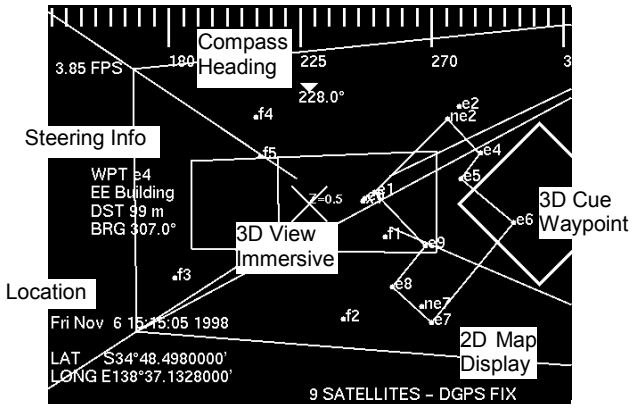

Figure 3 – 3D Immersive Display Example

Figure 3 shows the same 2D display as before, except a 3D immersive display of our office and whiteboard is now superimposed as well. When looking through this display, from a fixed sitting location, (without GPS) we were able to reasonably register the display with the outline of the office. In an outdoors use, we managed to make a box lock around a table as we walked around it.

The 3D display is still under development, as there are a lot of issues to do with accurately registering the image with the real world, and is the focus of a lot of our future research.

4 System Architecture

To support the navigation task, along with a wide range of AR and multimedia applications, we developed a highly modular to support this. The software system is broken up into various modules that communicate with each other using a connection-oriented protocol – in this implementation, TCP/IP.

4.1 Modular Approach

An example of an application specific module is

the navigation module, which reads waypoints from a database, along with position and heading information, to produce steering instructions for other modules to present to the user. The display module presents data from other modules in a graphical format to the user via the head up display.

The modular architecture supports many concepts such as data abstraction and hiding, collaboration with others, and the flexibility to plug new components in without modifications.

4.2 Communications

To interconnect modules, we used a client-server style architecture. The server is a data source for other modules, and it listens on a TCP/IP socket waiting for incoming requests from clients. A client that wishes to receive data will contact the server, and send a listen message to subscribe to it. Whenever the server updates the value of this data, it will send the new value out to all clients that have registered an interest in the message. A client receiving new data may use it to update the screen, or calculate new navigation parameters for example. Note that many servers in the system are actually clients for other servers as well.

The entire system operates asynchronously, and is data driven; if there is no new data in the system, no action will be taken by any of the software modules. To illustrate this, consider the case of a new incoming position from the GPS. The harvester will process the new data, and then distribute it to all clients. The navigation module will receive this update, and recalculate navigation information. The display module will eventually receive an update from the harvester, and the new steering instructions from the navigation module, and use these to redraw the screen to reflect the user's new location.

4.3 Software Modules

Apart from the standard display and hardware modules that form the core of the system, other kinds of modules have been written to extend the system's capability for various tasks.

Sound/watchdog modules – A process continuously monitors the various variables in the system, and will sound audible (beeps or wave sound files) and/or spoken (synthesized voice) alarms when it is detected that these values have exceeded some parameter.

Web module – This process interfaces to CGI programs run by the Apache web server, allowing users to find out information about the wearable computer from a web browser. The module provides information about the wearable location and orientation, and refreshes continuously on the screen. This is a good example of how the system can be interfaced to systems that are of different design.

DIS and LSAP module – The wearable computer contains an in-built object tracking module, which allows it to know the location of objects in the world, and follow their movements. An interface has been written to allow this tracker to share information (both ways) with the DIS (Distributed Interactive Simulation) and LSAP (Land Situational Awareness Picture System) based software used at DSTO. Using a pair of radio modems, the wearable is able to communicate from the field back to headquarters, and with this, it is possible for DSTO to produce simulated vehicles moving around a landscape, and to see these vehicles on the wearable head up display, presented as icons. Simultaneously, the user shows up on the simulation rendering software, so that others can see where the wearable is located. The ability of the wearable computer to share information and collaborate with other users (both mobile and stationary) improves the usefulness and applicability of the system.

5 Software Library

To implement the modular architecture, a software library to support this was designed, with goals being to be flexible, extendible, and layered. Layering was employed to provide increasing levels of abstraction for allowing modules to interact with the system at the appropriate level they require, while at the same time minimising code replication across the system, and localising possible errors. Rather than modules focusing on communicating with each other, the code only does the tasks it needs to do, and then makes calls to library functions that actually make connections, subscribe, and process new incoming data. As a result, writing software modules to fit into the system is simple, and with many of the low level and repetitive details hidden away, also quite small.

The libraries provide functionality for distributive processing, asynchronous I/O, dynamic configuration, and automatic code generation:

5.1 Running modules in parallel over TCP/IP

Each of the modules are implemented as separate Unix processes. This allows modules to be distributed over multiple processors on one machine, or multiple machines due to the network support. The ability for the system to support this at a fundamental level improves the scalability for larger, resource intensive applications. For example, the outdoor navigation system was distributed over both the laptop and a second 486 wearable, which was included to increase the limited I/O capabilities of the laptop.

5.2 Asynchronous I/O event handling

The core of the library revolves around an event handler which monitors open file descriptors and waits for them to become available for reading or writing. When data arrives on the socket, the data is read, processed, and then handed to the calling code. As the complexities of doing I/O are abstracted away from the calling code, (to the point where even the type of transport is not specified) it is straightforward for the TCP implementation to be replaced by UDP by simply rewriting the library code. Slow modem links requiring writes to be buffered, and support for

handling devices such as X servers, tty devices, and serial ports are integrated in already.

One interesting feature of the I/O library is the ability to plug in simulated devices. During testing, it was possible to not use the GPS or compass and still verify the other modules are working correctly.

5.3 Dynamic configuration from the DBMS

Most software tends to use statically compiled controls, or possibly a configuration text file. Our system takes configuration to the next level by loading all system parameters such as the location of modules, port numbers, device names, and screen colours into a series of relational database tables. When the software initialises, it queries the database and loads the values required. By sending messages throughout the system when changes are made, it is possible for clients to reconfigure themselves by querying the database. The software does not have to be restarted as would be required if the controls were static. The database proved to be very powerful because it can be changed remotely via the radio modems. A second feature is the strong type checking by the database engine (in our case PostgreSQL v6.4 [POST98]) rather than relying on parsing a text file.

This feature proved useful when performing testing outdoors, for example, tuning the various display options such as colours and font sizes.

5.4 Automatic message code generation

Due to the complexity of the software, (approaching 35,000 lines) maintenance is a major issue. For example, the protocol handlers, around 5000 lines, is responsible for reading in binary messages from the network, and converting them into C structures for compilation. At the same time, it will reorder fields and perform checking to allow different revisions of the system (with modified message formats) to talk with each other assuming they are reasonably similar. To do this, rather than writing source code, we wrote around 400 lines of definition files, which contain information about how the messages are formatted and the corresponding field names and types. A program known as STC (structure compiler) then writes source code for these files, so that when the definitions change, the compiler effects the changes, saving large amounts of time in rewriting.

7 Conclusion

This architecture designed at the University of South Australia Wearable Computer Lab has been designed from the outset to be an innovative and powerful way of implementing wearable computer systems. The architecture is data driven, and can benefit from distributing components as separate processes. A layered implementation eases application development, increases the overall reliability of the software, and gives a degree of device independence.

Our architecture has supported concept demonstrator applications such as terrestrial navigation, web-based visualisation, and integration with virtual reality systems.

These applications have been tested extensively outdoors, and show that the software architecture works, with good performance. During testing, frame rates of 20 – 30 fps (sufficient for smooth animation) have been achieved, with the main bottleneck being in processing and drawing the 2D and 3D data for the display. We are now investigating how this architecture will support geographical information system applications among others.

8 Acknowledgements

The authors would like to acknowledge the assistance of DSTO (Land Space Operations Division, Salisbury) for kindly providing the use of hardware for our research.

9 References

[AZUM97] R. Azuma, Survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4). 1997.

[BASS97] L. Bass, C. Kasabach, R. Martin, D. Siewiorek, A. Smailagic, J. Stivoric. The design of a wearable computer. In *CHI 97 Looking to the Future,* pp 139-146. ACM SIGCHI, ACM. 1997.

[MANN96] S. Mann, Smart Clothing: The Shift to Wearable Computing. In *Communications of the ACM*, Vol 39, No. 8, pp 23-24, Aug 1996.

[FEIN97] S. Feiner, B. MacIntyre, T. Hollerer, A. Webster. A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment. In *1st Intl. Symposium on Wearable Computers*, Cambridge, Ma, Oct, 1997. pp 74-81.

[PIEK98a] W. Piekarski, D. Hepworth, Outdoor Augmented Reality Navigation System – Project Documentation. University of South Australia, 1998.

[PIEK98b] W. Piekarski, D. Hepworth, V. Demczuk, B. Thomas, B. Gunther. A Mobile Augmented Reality User Interface for Terrestrial Navigation. In *Proc. of the 22nd Australasian Computer Science Conference*, Auckland, NZ, Jan 1999. pp 122-133

[POST98] PostgreSQL v6.4.2 Database Engine - http://www.postgresql.org, 1998.

[THOM97] B. Thomas, S. Tyerman, K. Grimmer, Evaluation of Three Input Mechanisms for Wearable Computers. In *1st Intl. Symposium on Wearable Computers*, Cambridge, Ma, Oct, 1997. pp 2-9.

[THOM98] B. Thomas, V. Demczuk, W. Piekarski, D. Hepworth, B. Gunther, A Wearable Computer System With Augmented Reality to Support Terrestrial Navigation. In *2nd Intl. Symposium on Wearable Computers*, Pittsburg, Pa, Oct 1998. pp 168-171.