REVIREN: Augmenting Virtual Environments with Personal Digital Assistants

RICHARD SCHRODER, MARC BOSMA, ANDREW COUSINS and GARY MARSDEN
University of Cape Town

Our research group is currently embarked on building cheap display-based virtual environments. With the use of Personal Digital Assistants becoming more widespread and commonplace, there is a need to investigate ways of allowing them to integrate better with our computing environment. We feel that these devices' versatility make them ideal as alternative input/output mechanisms for Virtual Environments.

By reducing peripheral information and removing display widgets from the main screen, we aim to manage screen real-estate and thus avoid obscuring the user's view of the environment. Our system makes use of a virtual shopping mall with various interactive elements designed to highlight the possibilities PDA-based control offers.

Categories and Subject Descriptors: H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*graphical user interfaces*; *input devices and strategies*; *user interface management systems*; B.4.2 [**Input/Output and Data Communications**]: Input/Output Devices

General Terms: Human factors, Design

Additional Key Words and Phrases: Mobile interface, Virtual environment control, Dynamic interface

1. INTRODUCTION

Our research group is currently embarked on building cheap display-based virtual environments. These environments are similar to those described by Bowman et al. [1998] in which the user can access information embedded within the environment. In our case, we are interested in developing a virtual shopping mall, where the user can select a particular item and be presented with information, such as price or size. In some sense this is similar to augmented reality, but this time the extra information is overlaid on a virtual environment rather than the real world.

A primary design goal when creating this type of environment is managing the screen real estate so that the information and widgets laid on top of the environment do not totally obscure the user's view of what lies behind. Solutions to this problem found in first-person three dimensional games rely on providing the user with information at the periphery of the screen. Other systems, such as Web Space [Mohageg et al. 1996], also make use of the screen periphery, but only to place common interaction widgets the user may wish to click on. Unfortunately, for our application, this approach is less suitable as the nature of the information and the task at hand vary greatly, as the following scenario illustrates:

> "A user wishes to check their bank balance. Using navigation keys on the screen, they move to the bank within the shopping mall. Once they arrive at the bank, their navigation buttons disappear and are replaced by a numeric keypad so that the user can enter their PIN. Once the PIN is entered successfully, these keys disappear and information about the account is displayed along with a widget

Author Addresses:
R Schroder, Department of Computer Science, University of Cape Town, Private Bag, Rondebosch, 7701, South Africa; rschrode@cs.uct.ac.za
M Bosma, Department of Computer Science, University of Cape Town, Private Bag, Rondebosch, 7701, South Africa; mbosma@cs.uct.ac.za
A Cousins, Department of Computer Science, University of Cape Town, Private Bag, Rondebosch, 7701, South Africa; acousins@cs.uct.ac.za
G Marsden, Department of Computer Science, University of Cape Town, Private Bag, Rondebosch, 7701, South Africa; gaz@cs.uct.ac.za

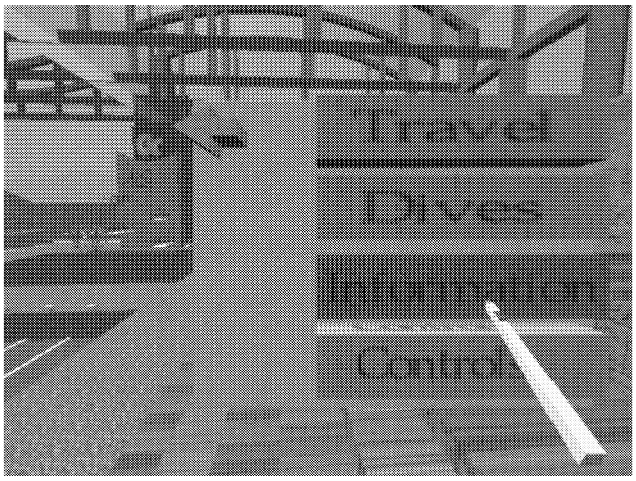

Figure 1. *Virtual "Pen and tablet" interaction as described by Bowman et al [1998]*

menu permitting further options for interacting with the bank account. The user then chooses to view their bank balance."

Whilst placing this information at the periphery may clear the main field of view, the constantly changing nature of this information means the display is distracting and hard to use. By removing the interaction widgets and display elements from the screen, the user's view of the environment is maintained.

2. PREVIOUS WORK

Previous attempts at augmenting virtual environments have focused on overlaying information or display widgets over the user's view of the world. The World-In-Miniature (WIM) metaphor described by Stoakley et al. [1995] makes use of this technique to display a scaled-down view of the user's world. Through this scaled-down model, users can rapidly move to or interact with elements in the virtual environment. The major drawback here is that the user's view becomes obscured by the model being overlaid.

Bowman et al. [1998] have developed a novel solution to this problem using head-mounted displays (HMDs) and tracking sensors. The user holds a physical clipboard that has a sensor attached. In the virtual view, this clipboard is used to display information and menu options. With this arrangement, the user can move the "overlay" in or out of their view as they wish. However, this solution makes use of expensive, dedicated hardware and is thus not a solution to our goal of creating cheap and effective displays.

For inspiration we returned to Bowman's paper where it was found that the most effective style of interaction with menus in a virtual environment was using a "pen and tablet" (see Figure 1). Rather than a virtual pen and tablet however, we decided to use a physical pen and tablet in the form of a Personal Digital Assistant (PDA) connected to the desktop machine running the VE.

Due to the increasing proliferation of PDAs, many computers now have these devices permanently attached for synchronisation and battery charging purposes. The PDA is also a highly flexible device as it supports output (to the screen), input (from hardware buttons or the touch screen) and can thus be programmed to provide a variety of interactive interfaces. This facility has already been explored in providing an interface to 2D-GUI based applications in Myers [2002]. We extended these ideas for use in conjunction with virtual environments. A large amount of work investigating the use of PDAs as an alternative form of input for 2D applications has been carried out at Carnegie Mellon University, resulting in the creation of the Pebbles Project [Pebbles Project]. This and other work involving interaction between PDAs and PCs has also mostly focused on the use of PDAs as a generic remote control, with limited or no feedback capabilities. To our knowledge, no work has been performed investigating the use of a PDA for explicitly controlling 3D environments.

The Pebbles project has resulted in the development of the Personal Universal Controller (PUC) system [Nichols et al. 2002]. This is also a generic remote control solution, but with the distinction being that the device is self-programming. The PUC "discovers" devices that support it, retrieves their functionality set and generates an appropriate user interface using an XML-based specification language. This is a unique solution in that the result is a remote control which fully supports the device's functionality set, rather than a sub-set of common functionalities. Another advantage to the PUC is its support for multiple modalities – it may support both graphical and speech interfaces, switching between them where appropriate.

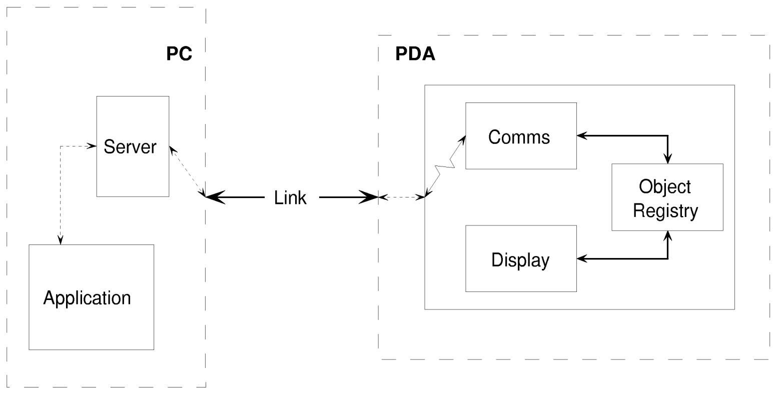

Figure 2. *Client/server structure of the system*

IBM's Universal Information Appliance (UIA) [Eustice et al. 1999] is similar to the PUC, but lacks the ability to program itself. Interfaces are specified using an XML-based meta-language called MoDAL (Mobile Document Application Language). Both the PUC and UIA systems dynamically generate interfaces to the devices they are controlling. However, when interfacing to a virtual environment, the designer needs explicit control over the interface layout. Thus, some way of specifying an interface is needed.

Jini [Jini], a Java-based specification developed by Sun was designed to make distributed computing simpler for networked devices. It provides a dynamic service discovery framework that allows devices to not only discover each other but to also discover services available and to signal events to each other [Newmarch 2003]. It uses a middleware-structured system, with a *client* contacting a *lookup service* to find *services* available for use. What the system doesn't do, however, is to provide any sort of explicit user interface specification – it simply allows devices to find one another and to utilise services offered. Whilst potentially useful for allowing the VE system to discover the mobile device, its Java-based nature makes it unsuitable due to the poor Java support offered on PDAs at present.

None of the user interface specification solutions presented here fully regard the remote device (or client) as a potential output device – it is simply treated as an alternative input device with a changeable interface. Given the multimedia capabilities of today's PDAs and other mobile devices, we feel this is a waste of available resources. So, rather than employ one of these rather limited standards to specify the interaction between computer and PDA, we decided to adopt the more general User Interface Markup Language (UIML) [UIML Standard]. This is an XML-based language that provides a highly device-independent method of describing user interfaces.

UIML is used to specify user interfaces in a platform- and device-independent syntax, allowing interface designers to concentrate on the interfaces without the worry of device intricacies. The language is a declarative, XML-compliant meta-language that originated in 1998. It is particularly useful for creating multiplatform, multimodal and multilingual dynamic user interfaces. This makes it an excellent choice for our system, allowing VE designers to specify the desired interface in a standardised manner, as well as taking into account any future technologies which may emerge.

3. SYSTEM DESIGN

The basic idea of our system is to render interaction widgets and environment information on the PDA rather than on the main display. Therefore, the environment authoring tool needs an Application Programming Interface (API) to communicate with the PDA and receive information back from it. This communication needs to be able to specify the interface (output) and to able to receive events (input). As mentioned in the previous section, we used UIML to specify these interfaces. Communication between the VE and the PDA was achieved through a basic client/server model (See Figure 2). The "server" element was added to the Genesis [Genesis 3D] tool we were using to create the shopping mall environment, allowing the virtual environment creator access to the UIML functionality of the PDA.

Early in development, it became clear that the PDA was not capable of rendering the different interfaces "on the fly." To solve this problem, we implemented a separate PDA proxy which ran as part of the server and pre-parsed all the XML files the PDA might need. Parsed interfaces can then be rapidly downloaded on demand in a simple script-based format. These parsed interfaces consisted of simple scripting commands which the PDA could then rapidly execute as it received them. Similarly, feedback from the PDA to the virtual environment used this script. Objects created on the PDA are then registered for events, such as taps or movement by the

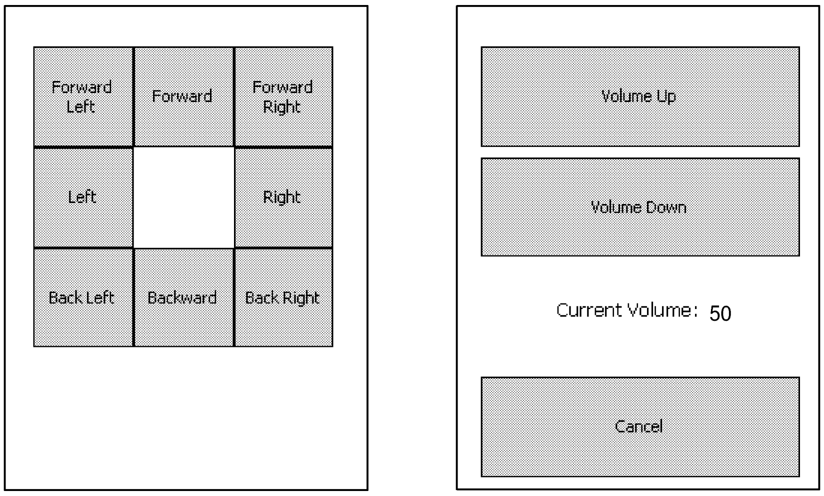

Figure 3. *Two scenarios of usage with the PDA: navigation and controlling volume levels in the sound room*

stylus on the PDA display. These events are the sources of the feedback the PDA provides. Objects can also be registered for more than one event simultaneously or even de-registered for specific events dynamically. This functionality would be useful for implementing a WIM-style scenario:

> “The user taps the WIM to select it. Whilst holding their finger or the stylus down, they rotate the model by dragging the model in the appropriate direction. Upon releasing the model, the object is de-selected and dragging has no effect.”

As an initial proof of concept, we decided to implement the most basic set of objects possible - labels for displaying text, buttons for retrieving input and lines and shapes for basic image drawing. This restriction was partly due to the slow nature of the PDA used, combined with the slowness of the serial connection to the PC. However, the design was left open for newer technologies such as Wireless networks or Bluetooth access. With more powerful PDAs now becoming available, multimedia objects such as images, animations and sound are now distinct possibilities. These objects would offer a richer and more interactive experience than simple text. With the ability to send multimedia objects to the device, visualisations such as the WIM metaphor become realities.

3.1 Examples of Interactions

The virtual shopping mall built using Genesis 3D contained three basic types of interaction over and above navigation control. (See Figure 4)

These interactions were included to illustrate the capabilities of the system. Basic one-way interaction was achieved through the control of an elevator by the user. When near or inside the elevator, a ninth button, coloured differently, appears in the center of the navigation control with the appropriate title (ie. “Call Lift” or “Use Lift”). Once the button has been triggered, it disappears and feedback is provided to the user by either the arrival or movement of the elevator.

A “sound room,” in which a continuous loop of music is played, was used to illustrate basic one-way interaction with simple feedback. Using the mobile device the user could control the volume level in the room, with the current volume level being displayed on the device’s screen. As the user changes the volume level (see Figure 3), either by tapping the appropriate button or holding the button down, the display is updated to show the current volume level.

The third form of interaction was complex two-way interaction between a user and an Automatic Teller Machine (ATM). Upon activating the ATM, the user is required to authenticate using a PIN code and can then perform one of three functions: check their balance, make a withdrawal or make a deposit. This forms a challenge-and-response style of interaction typical of most real-world ATMs.

Throughout these interactions, the only information displayed on the user’s view of the virtual environment was an icon to alert the user that the PDA display required their attention. This was located in the bottom left corner of the screen and did not obscure any of the user’s view. Navigation was achieved using interaction widgets on the PDA and the mouse to control the user’s view. No keyboard was used at any time during the interactions. The basic method of usage was for the user to manipulate the PDA with their non-dominant hand and the mouse with their dominant hand. This mimics the typical usage scenario of a user controlling a VE

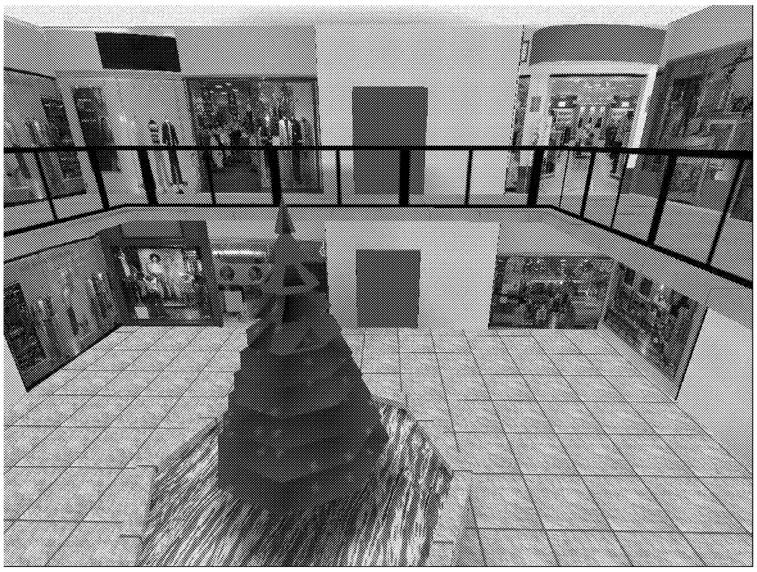

Figure 4. *A view inside the virtual mall*

using a keyboard/mouse combination, with the keyboard being operated by the non-dominant hand.

One problem that became apparent was the basic nature of a touch-screen – only one part of the screen may be engaged at any time. This made movements such as "forward-right" problematic. The solution to this problem was to replace a standard 4-way keypad with an 8-way keypad (see Figure 3). With the system able to dynamically add or remove elements, it raises interesting possibilities for indicating to the user what their options are. For example, if a user were standing with their back against a wall, all the "back" navigation buttons could be removed. In a similar vein, this system does not need to be restricted to VEs alone. Any system where interaction widgets or extra information needs to be displayed can make use of this set-up. Other potential uses of this system include tutorial-type situations, where instructions guiding the user could be displayed on the PDA, reducing screen clutter on the main display.

4. USABILITY

When considering the usability impact of the system, two viewpoints need to be considered. There is the effect of the system itself, and the effect of the manner that the system is used by the VE designer.

Reviren (REmote control for VIRtual ENvironments) is designed to aid the virtual environment experience through its provision of an alternate input/output interface. In *Why Interfaces Don't Work*, Norman makes an interesting point about users and software interfaces [Norman 1990]. He states that the problem with interfaces is that they are interfaces – users like to think of themselves as achieving a task, not as using a computer. Our system is specifically designed with the control of virtual environments in mind and can thus address the needs of the system directly without generality. Thus, following Norman's argument, the system will be able to perform unobtrusively, allowing the user to concentrate on the task at hand, rather than on how to perform the task.

Despite their excellent picking capabilities, touch screens have largely been dismissed as effective input mechanisms due to user fatigue [Downton and Jones 1993]. Not only was fatigue cited as an issue, but also that the user's preferred viewing distance is not always the same as their preferred pointing distance. However, a PDA resting on a surface in its cradle effectively eliminates these problems – the user's arm rests on the surface, as when using a mouse, and the PDA can then be held at a distance that is considered comfortable.

Lastly, the system should be used by the VE designer primarily to introduce flexibility to the VE interface. In this instance, flexibility is defined as the ability to achieve an objective or task in more than one manner [Downton and Jones 1993]. This allows the designer to cater for different classes of user, from the novice through to the expert user who makes frequent use of the system. As an example, the novice may not be able to remember keyboard mappings for interacting with the VE and may opt to use the PDA instead. Alternatively, the expert user may choose to forgo the PDA and use the keyboard. With a well designed system, both classes of user should be able to accomplish the same set of tasks given different input modalities.

Ideally, the system would undergo extensive user testing to verify the claims presented here. Unfortunately, time constraints prevented us from doing so and it would be the focus of any future work to take place on the system.

5. CONCLUSIONS

We have successfully built a remotely managed interface system on a PDA which allowed us to control a Virtual Environment. The system as it stands only supports a basic set of object types and expanding this set of objects should be the focus of future work. Other future work on this system should include a detailed usability study to investigate its usefulness to both novice and experienced users.

This system enhances the interactivity of VEs, removing the need for users to recall keyboard shortcuts and other commands. We feel that it leads to better management of screen real-estate, making the VE less distracting and less cognitively demanding for the user.

ACKNOWLEDGMENTS

This work was supported by an equipment donation from Microsoft.

REFERENCES

Bowman, D. A., Hodges, L. F., and Bolter, J. 1998. The virtual venue: User-computer interaction in information-rich virtual environments. *Presence 7,* 5 (Oct.), 478–493.

Downton, A. and Jones, E. 1993. *Engineering the Human-Computer Interface.* McGraw-Hill. ISBN 00770772.

Eustice, K. F., Lehman, T. J., Morales, A., Munson, M. C., Edlund, S., and Guillen, M. 1999. A universal information appliance. *IBM Systems Journal 38,* 4, 259–266.

Genesis 3D. http://www.genesis3d.com/ (Accessed March 2004).

Jini. Jini Specifications and API Archive. http://java.sun.com/products/jini/ (Accessed July 2004).

Mohageg, M., Myers, R., Marrin, C., Kent, J., D.Mott, and Isaacs, P. 1996. A user interface fpr accessing 3d content on the world wide web. In *Proceedings of the SIGCHI.* ACM Press, 466–472.

Myers, B. 2002. Mobile devices for control. In *Proceedings of the 4th International Symposium on Mobile HCI.* Springer, 1–8.

Newmarch, J. September 2003. Jan newmarch's guide to jini technologies. http://pandonia.canberra.edu.au/java/jini/tutorial/Jini.xml.

Nichols, J., Myers, B., Higgins, M., Hughes, J., Harris, T. K., Rosenfeld, R., and Pignol, M. 2002. Generating remote control interfaces for complex appliances. In *Proceedings of the 15th ACM Symposium of User Interface Software and Technology.* ACM Press, 161–170.

Norman, D. A. 1990. *The Art of Human-Computer Interface Design.* Addison-Wesley, Chapter Why Interfaces Don't Work, 209–219. ISBN 0201517973.

Pebbles Project. http://www.cs.cmu.edu/~pebbles/ (Accessed March 2004).

Stoakley, R., Conway, M. J., and Pausch, R. 1995. Virtual reality on a wim: Interactive worlds in miniature. In *Proceedings of the SIGCHI.* ACM Press / Addison-Wesley, 265–272.

UIML Standard. http://www.uiml.org/ (Accessed March 2004).